**Министерство науки и высшего образования РФ**

**Автономная некоммерческая организация высшего образования**

**Самарский университет государственного управления**

**«Международный институт рынка»**

**Институт экономики и права**

**Кафедра информационных систем и компьютерных технологи**

**Направление 09.03.03 прикладная информатика**

**ДОПУСКАЕТСЯ К ЗАЩИТЕ**

Заведующий кафедрой:

д.т.н., профессор Макаров А.А.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА БАКАЛАВРА**

**«Разработка корпоративного портала по производству мороженного САМПО»**

Выполнил: Крайнов В. А., гр. Пи-41  
**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Научный руководитель:

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Оценка:

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Самара 2023

# Аннотация

Выпускная квалификационная работа состоит из корпоративного сайта (НАПИСАТЬ ИМЯ САЙТА) и пояснительной записки на (НАПИСАТЬ СКОЛЬКО СТРОНИЦ) страницах, содержащей (НАПИСАТЬ СКОЛЬКО РИСУНКОВ) рисунка, (НАПИСАТЬ СКОЛЬКО ТАБЛИЦ) таблиц, (НАПИСАТЬ СКОЛЬКО ИСТОЧНИКОВ) источников литературы, а также (НАПИСАТЬ СКОЛЬКО ПРИЛОЖЕНИЙ) приложение.

**Объектом** выпускной квалификационной работы является деятельность общества с ограниченной ответственностью (ООО) «**ФАБРИКА МОРОЖЕНОГО САМ -ПО**», занимающийся производством мороженного.

**Предметом** выпускной квалификационной работы является сайт компании (ООО) «**ФАБРИКА МОРОЖЕНОГО САМ -ПО**», который создается с целью привлечения новых клиентов и увеличения доли, занимаемой компанией на рынке.

**Целью** выпускной квалификационной работы является создание корпоративного web-сайта для компании (ООО) «**ФАБРИКА МОРОЖЕНОГО САМ -ПО**».

В соответствии с поставленной целью в работе определены следующие **задачи:**

• проведен анализ предметной области;

• проанализированы существующие корпоративные сайты компаний конкурентов;

• проведен анализ средств для разработки корпоративного сайта;

• разработана структура корпоративного сайта;

• сайт наполнен контентом. Результатом работы будет являться корпоративный сайт компании (ООО) «**ФАБРИКА МОРОЖЕНОГО САМ -ПО**».

# Обозначения сокращений

* JS – JavaScript;
* БД-База данных
* IDE-Интегрированная среда разработки
* DOM- объектная модель документа
* SWOT -  [метод](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4) [стратегического планирования](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D0%B0%D1%82%D0%B5%D0%B3%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%BE%D0%B5_%D0%BF%D0%BB%D0%B0%D0%BD%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)
* CMS - Система управления содержимым
* UX/UI- Дизайн взаимодействия с пользователем
* Frontend - интерфейс
* Backend- серверная часть
* Верстка- создание структуры странице на разметке html
* Тег (html-тег)-управляющая символьная последовательность
* Sсript- сценарий
* framework - программная платформа, определяющая структуру программной системы

# Содержание

Оглавление

[Аннотация 2](#_Toc136819632)

[Обозначения сокращений 4](#_Toc136819633)

[Содержание 5](#_Toc136819634)

[Введение 5](#_Toc136819635)

[1. Теоретические основы использования корпоративных портальных технологий 8](#_Toc136819636)

[1.1 Функции корпоративных порталов 8](#_Toc136819637)

[1.2 Назначение корпоративных порталов 9](#_Toc136819638)

[1.3 Классы корпоративных порталов 10](#_Toc136819639)

[1.4 Анализ языков программирования 10](#_Toc136819640)

[1.5 Анализ систем IDE 19](#_Toc136819641)

[2. Аналитическая часть 25](#_Toc136819642)

[2.1 Характеристика компании 25](#_Toc136819643)

[2.2 Постановка проекта разработки сайта компании 25](#_Toc136819644)

[2.3 Анализ текущего состояния web-сайта компании 26](#_Toc136819645)

[2.4 Обоснование проектных решений по видам обеспечения 27](#_Toc136819646)

[3. Реализация разработанного корпоративного портала компании 29](#_Toc136819647)

[3.1 Этапы разработки 29](#_Toc136819648)

[3.2 Технологии, использованные для разработки сайта 72](#_Toc136819649)

[ЗАКЛЮЧЕНИЕ 72](#_Toc136819650)

[СПИСОК ЛИТЕРАТУРЫ 72](#_Toc136819651)

# Введение

Ни для кого не секрет, что для успешного развития компании нужны клиенты. А значит, возникает задача поиска клиентов и необходимость, в том, чтобы о компании узнали люди. В данном случае оптимальным решением данной проблемы может стать создание собственного сайта.

Главной задачей профессионально построенного сайта компании является превращение посетителя, зашедшего на сайт, в потенциального клиента.

Объект исследования дипломной работы – разработка web-сайта компании.

Предмет исследования дипломной работы – web-сайт компании, оказывающей услуги по продаже мороженного.

**Объектом** выпускной квалификационной работы является деятельность общества с ограниченной ответственностью (ООО) «**ФАБРИКА МОРОЖЕНОГО САМ -ПО**», занимающийся производством мороженного.

**Предметом** выпускной квалификационной работы является сайт компании (ООО) «**ФАБРИКА МОРОЖЕНОГО САМ -ПО**», который создается с целью привлечения новых клиентов и увеличения доли, занимаемой компанией на рынке.

**Целью** выпускной квалификационной работы является создание корпоративного web-сайта для компании (ООО) «**ФАБРИКА МОРОЖЕНОГО САМ -ПО**».

В соответствии с поставленной целью в работе определены следующие **задачи:**

• проведен анализ предметной области;

• проанализированы существующие корпоративные сайты компаний конкурентов;

• проведен анализ средств для разработки корпоративного сайта;

• разработана структура корпоративного сайта;

• сайт наполнен контентом. Результатом работы будет являться корпоративный сайт компании (ООО) «**ФАБРИКА МОРОЖЕНОГО САМ -ПО**».

# 1. Теоретические основы использования корпоративных портальных технологий

## 1.1 Функции корпоративных порталов

EIP (Enterprise Information Portal) - Корпоративный информационный портал обеспечивает персонифицированный доступ к внутренним и внешним информационным ресурсам компании. Обеспечивает доступ к данным по классификатору и с возможностью проведения сквозного полнотекстового и атрибутивного поиска.

EEP (Enterprise Expertise Portals) – Корпоративный экспертный портал, обеспечивает подключение к экспертам (связь между пользователями) на основе их знаний (способностей). Не всегда портал содержит нужную пользователю информацию, однако, в случае ее отсутствия, он может подключить пользователя к нужным специалистам, которые могут поделиться необходимой информацией, дать экспертные оценки по интересующим вопросам.

EAP (Enterprise Application Portal) - Корпоративный портал приложений. Портал, который предоставляет пользователям различные корпоративные приложения компании, с помощью которых они могут получить требуемую информацию и данные.

ECP (Enterprise Collaboration Portal) - Корпоративный портал совместной работы. Портал, предоставляющий информацию и приложения, и обеспечивающий работу группы сотрудников над какой-либо задачей, проектом (фактически, автоматизация бизнес-процессов).

EKP (Enterprise Knowledge Portal) - Корпоративный портал управления знаниями. Интегрированный портал, охватывающий все перечисленные выше, предназначенный для обеспечения коллективной работы сотрудников с максимальным информационным обеспечением и аккумулированием сотрудников знаний в виде корпоративной информации.  
Первые три типа корпоративных порталов это порталы для индивидуальной работы, они обеспечивают пользователя необходимой ему информацией, приложениями, экспертными заключения специалистов (которых он может найти и к которым пользователь может подключиться с помощью портала). Иногда все их объединяют под общим названием «Корпоративный информационный портал».

Вторые два типа корпоративных порталов это порталы совместной работы. Они позволяют проводить определенный цикл работ в соответствии с алгоритмом бизнес-процессов, автоматически передавать промежуточные результаты по цепочке другимпользователям, отслеживать и координировать ход работ. При этом пользователи имеют максимальное информационное обеспечение, а компания аккумулирует корпоративную информацию, превращая ее в свой капитал.

## 1.2 Назначение корпоративных порталов

Перечисленные ниже типы называют и как корпоративные порталы, и как системы.

B2C (Business-to-Consumer) - Автоматизированные порталы (системы) электронной коммерции, в которых в качестве продавца выступает юридическое лицо (предприятие, организация), а покупателя - физическое лицо. Типичный пример - Интернет-магазин.

B2B (Business-to-Business) - Автоматизированные порталы (системы) электронной коммерции, в которых в качестве субъектов процессов продажи и покупки выступают юридические лица (предприятия, организации). Такие системы (электронные торговые площадки) обычно используются для организации снабжения компаний и реализации продукции.

B2E (Business-to-Employee) - Внутрикорпоративный портал (система), по организации работы персонала компании, отдельных структур или подразделений. Фактически, B2E системы являются системами автоматизации управления и берут на себя некоторые функции Корпоративных информационных систем (КИС). Как правило, такие системы функционируют во внутрикорпоративной сети - Интранет. При этом Интранет может использовать защищенные каналы в Интернет.

## 1.3 Классы корпоративных порталов

EAIP - (Enterprise Application Integration Portal) – Корпоративный портал служащий для интеграции разнородных корпоративных систем (КИС) компании. Следует сразу отметить, то, что применение термина EAIP нам неизвестно, и что интеграция корпоративных систем (EAI - Enterprise Application Integration) возможна и другими путями, не только применением технологии корпоративных порталов.

EAIP - (в отличие от EAP) осуществляет полную интеграцию всех разнородных корпоративных систем компании на основе единого пользовательского интерфейса. К EAIP легко подключать системы, имеющие Web-интерфейс, однако в большинстве случаев, как правило, такого интерфейса нет, для их подключения в портал приходится создавать интеграционные компоненты. В каждом таком случае речь фактически идет о написании двух слоев логики: нового Web-интерфейса старого приложения и слоя обмена данными с этим приложением.

Главный плюс корпоративного портала EAIP —его простота. Главный минус — это то, что по сути своей приложения остаются не интегрированными, а бизнес-процесс не становится сквозным. Доступ к приложениям для пользователя становится проще, и это все.

## 1.4 Анализ языков программирования

Во-первых, вам нужно выбрать, какие языки программирования и редакторы кода нам необходимо использовать в процессе разработки проекта.

Существует несколько языков программирования для разработки:

• JavaScript;

• Java;

• PHP;

• Python;

• Ruby;

• Perl;

• C;

• C++;

Необходимо проанализировать литературу по каждому языку веб-программирования, знать его характеристики, преимущества и недостатки и выбрать именно тот язык, который подходит для данного проекта. Рассмотрим самые популярные на данный момент:

**JavaScript**

Первоначально он был создан, чтобы сделать веб-страницы «живыми». Программы на этом языке называются скриптами. В браузере они напрямую связаны с HTML и выполняются сразу после загрузки страницы.

Программы JavaScript представляют собой обычный текст. Они не требуют специальной подготовки.

Преимущества:

* Быстро для конечного пользователя: JavaScript написан на стороне клиента, поддержка веб-сервера не требуется. Его также не нужно компилировать на стороне клиента, что дает ему некоторые преимущества в скорости. Поскольку скрипт выполняется на компьютере пользователя, в зависимости от поставленной задачи результаты практически немедленные. Например, вы можете проверить ввод пользователя перед отправкой запроса на сервер. Это снижает нагрузку на сервер.
* Простота: JavaScript относительно прост в изучении и реализации. Он использует модель DOM, которая предоставляет множество предопределенных функций для различных объектов на страницах, что упрощает создание сценариев для пользовательских целей.
* Возможность использования: JavaScript прекрасно работает на других языках, может быть использован в самых различных приложениях, а также может быть использован для различных приложений. Сейчас существует много способов использовать JavaScript через Node серверы. js. Если node загрузил пользователь. js посредством Express использует документовую базу, например, Mongodb и JavaScript внешним интерфейсом для клиента, а пользователь с помощью JavaScript может создавать приложение JavaScript в основном окне с использованием лишь JavaScript.

Недостатки:

* Безопасность: JavaScript явно добавляется на веб-страницы и в клиентские браузеры, он может использовать систему пользователя, позволяя запускать вредоносный код на клиентском устройстве.
* Поддержка браузера: JavaScript иногда по-разному интерпретируется разными браузерами. Различные механизмы компоновки могут отображать JavaScript по-разному, что приводит к несоответствиям с точки зрения функциональности и пользовательского интерфейса. Большая часть JavaScript зависит от манипулирования элементами DOM браузера. И разные браузеры предоставляют разные способы доступа к объектам, особенно Internet Explorer.
* Растущее число конкурентов: JavaScript — очень старый язык сценариев, который работает на машинах, и есть другие технологии, которые делают то же самое на его месте (например, JQuery), но лучше и проще.
* Если вы отключите выполнение JavaScript в своем браузере, весь код JavaScript не будет выполняться.
* Загрузка файла: файл JavaScript загружается на клиентский компьютер, чтобы любой мог прочитать код и использовать его повторно.

**Java**

Java — объектно-ориентированный язык программирования, разрабатываемый компанией Sun Microsystems с 1991 года и официально выпущенный 23 мая 1995 года. Первоначально новый язык программирования назывался Oak (Джеймс Гослинг) и разрабатывался для бытовой электроники, но позже был переименован в Java. и начал использоваться для написания апплетов, приложений и серверного программного обеспечения.

Java-программы могут быть преобразованы в байт-код, который запускается на виртуальной машине Java (JVM) — программе, которая обрабатывает байт-код и передает инструкции оборудованию, подобно интерпретатору, но с той разницей, что байт-код, в отличие от текста, обрабатывается намного быстрее.

Виртуальная машина Java (JVM) используется для запуска Java. JVM обрабатывает байт-код, после чего процессор обрабатывает код, полученный от JVM. Все виртуальные машины работают одинаково, поэтому один и тот же код работает одинаково во всех операционных системах, что делает Java кроссплатформенным языком программирования.

Преимущества:

* Java — кроссплатформенный язык. Код Java, написанный для одной платформы (то есть операционной системы), может работать на других платформах без изменений.
* Java делает общедоступными свои апплеты (апплеты) — небольшие, надежные, динамичные и независимые от платформы активные сетевые приложения, встроенные в веб-страницы Java-апплеты, можно настраивать и распространять среди потребителей так же легко, как и любой документ HTML.
* Java раскрывает возможности разработки объектно-ориентированных приложений, сочетая простой и знакомый синтаксис с надежной и простой в использовании средой разработки. Это позволяет широкому кругу программистов быстро создавать новые программы и новые апплеты.
* Java предоставляет программисту богатый набор классов объектов, чтобы четко абстрагироваться от многих системных функций, используемых в окнах, сетях и ввода-вывода. Основная особенность этих классов заключается в том, что они предоставляют независимые от платформы абстракции для широкого спектра системных интерфейсов.

Недостатки:

* Низкое, по сравнению с другими языками, быстродействие, повышенные требования к объему оперативной памяти (ОЗУ).
* Большое количество библиотек и стандартных технологий создают трудности в изучении языка.
* Постоянное развитие языка обуславливает наличие устаревших и новых инструментов, имеющих одинаковое функциональное назначение. Там же были перечислены некоторые особенности языка.
* Java — полностью объектно-ориентированный язык. Например, C++ также является объектно-ориентированным, но он позволяет писать программы в необъектно-ориентированном стиле, чего Java не может.
* Реализовано с использованием интерпретации R-кода (байт-кода). Те. программа сначала транслируется в машинно-независимый P-код, а затем интерпретируется некоторой программой-интерпретатором (Java Virtual Machine, JVM).

**PHP**

PHP известен как серверный язык программирования. Это означает, что он работает на веб-сервере. Большинство языков веб-программирования являются языками на стороне сервера, но некоторые, например, JavaScript, работают на стороне клиента, что означает, что они работают в веб-браузере.

Выполнение кода на стороне сервера более безопасно, чем на стороне клиента, как и JavaScript. Поскольку код JavaScript отправляется в веб-браузер, посетителям сайта легко просматривать и редактировать его. Даже на одной странице сайта вы можете легко комбинировать PHP и JavaScript. Серверный код остается на веб-сервере и недоступен посетителям сайта. PHP — это инструмент, который находится на веб-сервере и запускает там PHP-скрипты.

PHP — это программное обеспечение с открытым исходным кодом. Это означает, что любой пользователь может получить доступ и работать с PHP. Это помогает гарантировать, что PHP будет работать в течение длительного времени. PHP можно загрузить и использовать бесплатно. Именно по этой причине многие хостинг-провайдеры широко используют PHP. Пользователь обнаружит, что подавляющее большинство веб-хостов поддерживают работу PHP.

Одной из замечательных особенностей PHP является то, что программист может включать блоки кода PHP в HTML-страницы.

Пользователь может изолировать блоки PHP специальными символами. Когда веб-сервер получает информацию о странице, все блоки PHP запускаются движком PHP, в то время как другие части страницы отправляются браузеру «как есть».

Эта функция позволяет легко сделать обычные веб-страницы интерактивными. Отличный инструмент для контактных форм и форм с аналогичными функциями.

Достоинства:

* Это бесплатное программное обеспечение, распространяемое по специальной лицензии (лицензия PHP).
* Легко учиться на всех этапах.
* Поддерживается большим сообществом пользователей и разработчиков.
* Имеет расширенную и пополняемую поддержку базы данных.
* К языку имеется большое количество библиотек и расширений.
* Может использоваться в изолированной среде.
* Предоставляет собственные инструменты для организации веб-сессий, API расширения.
* Это довольно полная замена проприетарной среде Microsoft Active Server Pages (ASP).
* Можно развернуть практически на любом сервере.
* Портирован на большое количество аппаратных платформ и операционных систем.

Недостатки:

* Для создания настольного приложения или системного компонента не подходит.
* Он имеет слабые средства работы с исключениями.
* Глобальные конфигурационные параметры влияют на основной языковой синтаксис, что мешает настроить сервер и запускать приложения.
* По значению объекты передаются, что беспокоит многих разработчиков, которые привыкли передачу объектов ссылками, как это происходит на большинстве языков других.
* Приложения веб-приложений, написанные в PHP, часто страдают от проблем безопасности.

**Python**

Python широко используется как язык интерпретации скриптов разного назначения, хотя есть и переводчики языка Python.

Python предназначено для того, чтобы приблизить синтаксис реальных программ, написанных на них, к описанию задачи псевдокодов, позволяющей программисту сократить объем программы. Идея о создании этого языка появилась в конце 80-х годов и реализовалась Гвидо ван Россом.

Элегантное оформление и эффективный, дисциплинированный синтаксис языка позволяют программистам заниматься совместной работой по коду. Python является мультипарадигмальным языком программирования: позволяет сочетать процедурный подход в написании кода и объектно-ориентированный, и функциональный.:

Достоинства:

* Открытые разработки.
* В изучении достаточно просто, особенно в начальной стадии.
* Синтаксис и синтаксис стимулируют программистов писать хороший код.
* Предлагает средства быстрых прототипов и динамических семантик.
* Имеет огромное сообщество, которое позитивно настроено на новичков.
* Многие полезные библиотеки и языковые расширения могут легко использоваться в проектах, благодаря полностью унифицированному импортному механизму и программному интерфейсу.
* Модульные механизмы хорошо проработаны и можно легко использовать.
* В Python абсолютно всё - объекты в понимании объектно-ориентированного программирования.

Недостатки:

* Не слишком удачно поддерживать многопоточность.
* Python создает не так много качественных проектов, как другие универсальные языки программирования, такие как Java.
* Это не коммерческая поддержка средств разработки, хотя ситуация меняется со временем.
* Изначальное ограничение средств на работу с базами.
* Бенчмарки демонстрируют меньший производительность Python в сравнении с основными VM Java, что делает язык медленным.

Вывод: взвесив все плюсы и минусы языков я выбрал язык JavaScript, благодаря его гибкости от будет подходить под frontend и backend разработку

## 1.5 Анализ систем IDE

Существует несколько текстовых редакторов:

• Sublime Text;

• PyCharm;

• IntelliJ IDEA;

• Spyder;

• Eclipse;

• NetBeans;

• Komodo;

• Visual Studio Code;

Необходимо проанализировать литературу по каждому редактору, знать его характеристики, преимущества и недостатки и выбрать именно тот редактор, который подходит для данного проекта.

**Sublime Text**

Преимущества:

* большой выбор дополнений для поддержки синтаксиса и редактирования кода;
* навигация Goto Anything дает возможность быстро получать доступ к файлам;
* позволяет быстро перемещаться между проектами с сохранением изменений;
* множественное выделение позволяет вносить изменения в строки кода сразу в нескольких местах.

Недостатки:

* для полного использования всех функций необходимо приобрести лицензии;
* отсутствует функция автоматической отладки и боковая панель, как, например, в Visual Studio Code.

**PyCharm**

Преимущества:

* авто завершение кода, автопоиск и исправление ошибок в коде;
* поддержка сред для веб-разработки Django, Flask, Google App Engine, Pyramid и web2py;
* интеграция с системами контроля версий;
* интеграция с Jupyter Notebook, Anaconda и другими библиотеками;
* отладчик, профилировщик Python, встроенные терминал и инструменты баз данных.

Недостатки:

* версия Community не поддерживает многие инструменты, нет профилировщика Python и поддержки баз данных;
* отсутствует возможность оплаты Professional версии в России.

**IntelliJ IDEA**

Преимущества:

* авто завершение и исправление кода;
* мультиязычность;
* удобная рабочая среда, которая интуитивно понимает ваши действия и автоматически вызывает нужные инструменты;
* интеграция с системами контроля версий;
* анализ кода в реальном времени и надежные рефакторинги.

Недостатки:

* отсутствие в Community-версии инструментов профилирования, баз данных;
* отсутствует возможность удаленной разработки;
* сложная платформа для начинающих разработчиков.

**Spyder**

Преимущества

* бесплатная интегрированная среда для разработки;
* возможность просмотра значений переменных;
* доступная функция анализа ошибок с подсказками в случае нахождения ошибок;
* авто дополнение кода;
* кроссплатформенность, дает возможность работать на Windows, macOS, Linux.

Недостатки

* инструмент во многом предназначен для работы с данными. Для других задач подходит не так хорошо, как другие IDE.

**Eclipse**

Преимущества:

* бесплатная опенсорс среда;
* мультиязычность;
* доступны расширение возможностей и тонкая настройка за счет дополнительных плагинов;
* вокруг IDE сформировано большое комьюнити, помогающее с решением возникающих проблем;
* настраиваемый интерфейс;
* возможность интеграции JUnit и проведения оптимизации тестов.

Недостатки:

* возможности Eclipse не такие большие, как у Microsoft Visual Studio, но она распространяется с открытым исходным кодом, поэтому можно настраивать под свои потребности;
* сложная IDE для разработчиков-новичков.

**NetBeans**

Преимущества:

* доступна возможность установки пакетов для работы с другими языками;
* умное авто завершение кода и визуальный отладчик;
* простой и настраиваемый интерфейс рабочего пространства;
* возможности совместной разработки при работе в команде.

Недостатки:

* большое потребление памяти.

**Komodo**

Преимущества:

* авто завершение кода и интеллектуальная подсветка синтаксиса;
* визуальный отладчик;
* мультиязычность;
* интеграция с системами управления версиями;
* для физических лиц версия бесплатная;
* позволяет кастомизировать рабочее пространство;
* имеет визуальный отладчик и инструменты тестирования.

Недостатки:

* для работы в команде или на enterprise уровне есть платная версия.

**Visual Studio Code**

 Преимущества:

* кроссплатформенность: поддержка на Windows, macOS и Linux;
* встроенная технология авто дополнения кода IntelliSense;
* множество бесплатных расширений в маркетплейсе;
* полностью бесплатный редактор.

Недостатки:

* в Visual Studio Code доступно меньше инструментов, чем в полноценной IDE от Microsoft;
* отсутствует поддержка split screen.

**Вывод**: взвесив все плюсы и минусы языков я выбрал редактор Visual Studio Code, благодаря его гибкости, кроссплатформенность и множеству бесплатных расширений от будет подходить под frontend и backend разработку

# 2. Аналитическая часть

## 2.1 Характеристика компании

Фабрика мороженого «САМ-ПО» была основана при участии знаменитой датской фирмы «Тетра Пак Хойер» и финской компании «Хуура» в 1994 году. Удачное сочетание европейских технологий и трудолюбия российских специалистов позволило в намеченные сроки завершить строительство фабрики и 5 июля 1995 года была выработана первая продукция. Затем было проведено несколько реконструкций, которые затронули основные технологические циклы. На сегодняшний день мощность производства составляет 100 тонн мороженого в сутки, которое вырабатывается на самом передовом североамериканском и европейском оборудовании.

Сегодня фабрика мороженого «САМ-ПО» - одно из современнейших высокотехнологичных предприятий России. Почти 80 % мороженого вырабатывается по ГОСТу и не содержит растительных жиров. И это на протяжении всей истории фабрики является концепцией. Что касается ассортимента, то такие товарные марки, как «Кум- Ваня», «Кума-Маня», «Гигант из вафельной страны», «Ёжик из вафельной страны», «На набережной», «ГОСТ», «Сам-поЛакомься» были придуманы и созданы специалистами и стали нарицательными и «брэндовыми» в отрасли

## 2.2 Постановка проекта разработки сайта компании

2.2.1 Цель и назначение создания веб сайта

* Увеличение прибыли за счет привлечения новых клиентов
* Повышение конкурентоспособности компании
* Увеличение доли рынка
* Повышение барьеров входа на рынок для новых предприятий конкурентов
* Повышение удобства сайта для пользователей и увеличение конверсии
* сокращение объема затрат на наружную рекламу и рекламу в печатной продукции, за счет охвата аудитории по средствам сети Интернет

## 2.3 Анализ текущего состояния web-сайта компании

2.3.1 Информационная модель и ее описание

Так как web-сайт компании создан с целью привлечения клиентов, его функционал позволяет пользователям просматривать необходимую информацию об услугах и ценах, читать новости, получить контактную информацию компании. Для администратора web-сайт позволяет изменять информацию, размещенную на сайте, управлять навигацией на сайте, изменять формы на сайте.

2.3.2 SWOT-анализ

В ходе выпускной квалификационной работы был проведен SWOTанализ, который позволил выявить сильные и слабые стороны предприятия, а также угрозы и возможности внешней среды. Результаты представлены в таблице 1.

Таблица 1 — SWOT–анализ предприятия до создания сайта.

|  |  |
| --- | --- |
| Сильные внутренние стороны | Слабые внутренние стороны |
| 1. Постоянные лояльные клиенты.  2. Узнаваемость в своем городе. | 1. Отсутствие рекламы.  2. Ограниченность деятельности в связи с территориальным расположением. |
| Возможности во внешней среде | Угрозы внешней среды |
| Возможности использования оффлайн рекламы | 1. Снижение цен у конкурентов.  2. Изменение госстандартов и законодательства. |

SWOT-анализ после создания сайта представлен в таблице 2.

Таблица 2 — SWOT–анализ предприятия после проведения работ по переработке сайта и SEO-оптимизации

|  |  |
| --- | --- |
| Сильные внутренние стороны | Слабые внутренние стороны |
| 1. Наличие полной информации о компании и продукции на сайте.  2. Появление клиентов из других городов. | Низкий рейтинг сайта с малым возрастом |
| Возможности во внешней среде | Угрозы внешней среды |
| 1. Возможность использования новых видов рекламы (интернет-рекламы). 2. Использование средств интернет продвижения сайта. | 1. Конкуренты уже имеют сайты с большим рейтингом.  2. Сильная конкуренция среди существующих компаний. |

## 2.4 Обоснование проектных решений по видам обеспечения

2.4.1 Обоснование выбора технического обеспечения

Техническим обеспечением проекта является сервер, постоянно находящийся в сети Интернет, предоставленный хостинг-провайдером и имеющий определенные настройки и производительность. Выбор хостинг провайдера и тарифа был произведен по следующим критериям:

* стоимость услуг хостинг-провайдера;
* наличие PHP версии 8.0 и выше;
* наличие СУБД MySQL-5.7 и выше;
* Appache-PHP-8
* объем выделенного места на жестком диске от 512 Мб

2.4.2 Обоснование выбора программного обеспечения

Выбор программного обеспечения при разработке сайта сводится к выбору CMS и ее модулей. При выборе программного обеспечения были учтены следующие критерии:

* бесплатная CMS;
* высокая степень надежности и защиты от взлома;
* возможность реализовать весь необходимый функционал сайта с использованием готовых модулей;
* удобство администрирования;
* возможность привязки макетов, сверстанных по стандартам языка гипертекстовой разметки (HTML) и каскадных таблиц стилей (CSS);
* возможность при возникновении в будущем необходимости быстрого расширения функционала системы;
* наличие русифицированного интерфейса;
* наличие русской документации и сообщества программистов

# 3. Реализация разработанного корпоративного портала компании

## 3.1 Этапы разработки

После постановки задач будущего сайта разрабатывается будущий шаблон или шаблоны, из которых можно выбрать нужный вариант. На данном этапе определяется, как будет выглядеть дизайн в целом, какие графические элементы будут использованы, а также какая структура будет у страниц и какие цветовые решения будут на нем присутствовать

Разработка дизайна сайта обычно занимает от нескольких дней до нескольких недель, очень сложные работы могут проводиться даже месяц

3.1.1 Разработка структуры Web-Сайта

Первое, на что обращает внимание человек, когда заходит на какой-либо веб-сайт, это его дизайн. Именно от того, насколько удачно сделан дизайн веб-сайта зависит захочет ли пользователь просматривать страницу дальше или закроет ее. Поэтому очень важно при проектировании веб-сайта хорошо понимать на какую аудиторию он рассчитан, какие потребности возникают у пользователей. Всеми этими вопросами занимается UX/UI.

Главная задача дизайнера – максимально повысить уровень удовлетворенности конечного пользователя от взаимодействия с продуктом.

Для создания каркаса можно воспользоваться специальным приложением или нарисовать от руки на бумаге в клеточку или в точечку, после чего отсканировать и положить в папку с материалами проекта

Логическая структура web-сайта представлена на рисунке 1.

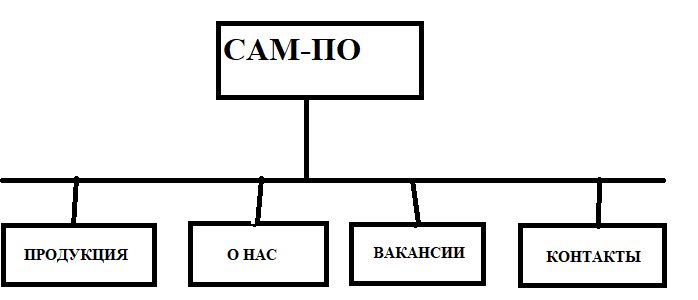


Рисунок 1 – Логическая структура web-сайта «САП-ПО»

Физическая структура web–сайта представлена на рисунке 2.

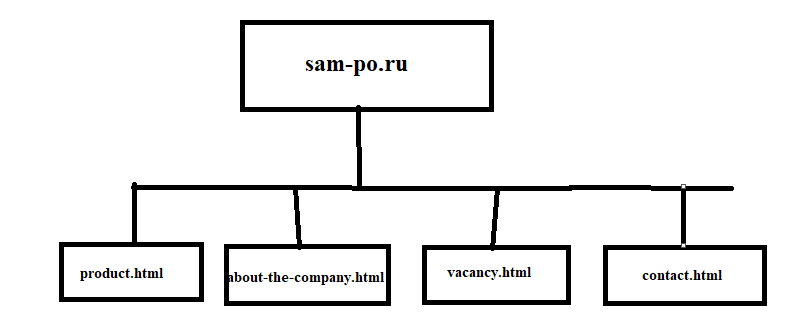


Рисунок 2 – Физическая структура web-сайта

В начале главной страницы web-сайта расположена шапка сайта и панель навигации. Следом идет основной контент страницы, содержащий информацию о продукции компании, а также 1 слайдер. Внизу страницы указаны контактные данные, информация об авторских правах.Блочная структура главной страницы web-сайта представлена на рисунке 3.

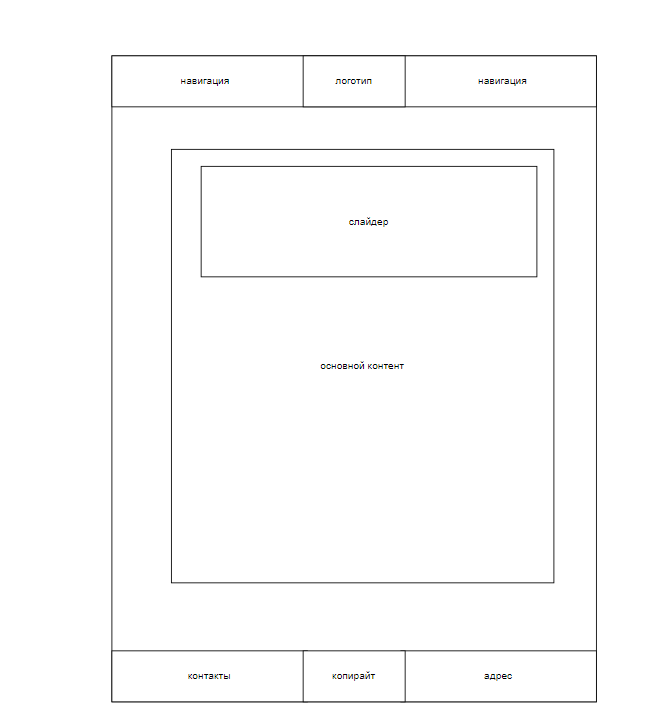


Рисунок 3 – Блочная структура главной страницы

На страницах с информацией, так же, как и на главной, вверху расположена шапка с логотипом и навигацией. Затем следуют блоки с подробной информацией. В самом низу страницы расположен «подвал» страницы с указанием контактных данных, информации об авторских правах. Блочная структура страниц, с описанием продукции на рисунке 4.

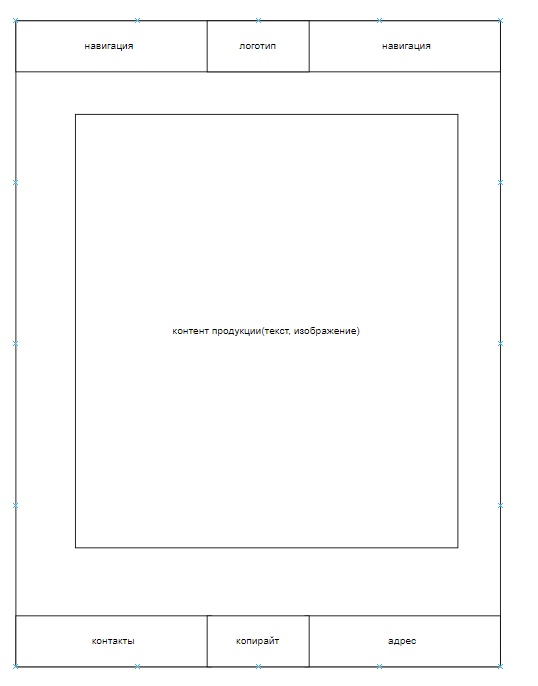


Рисунок 4 – Блочная структура страницы с описанием продукции

Страница «О нас» практически идентичны главной странице, за исключением отсутствия на ней блока со слайдером. На данной странице присутствует полноценная информация о компании. Блочная структура страниц «О нас» на рисунке 6.

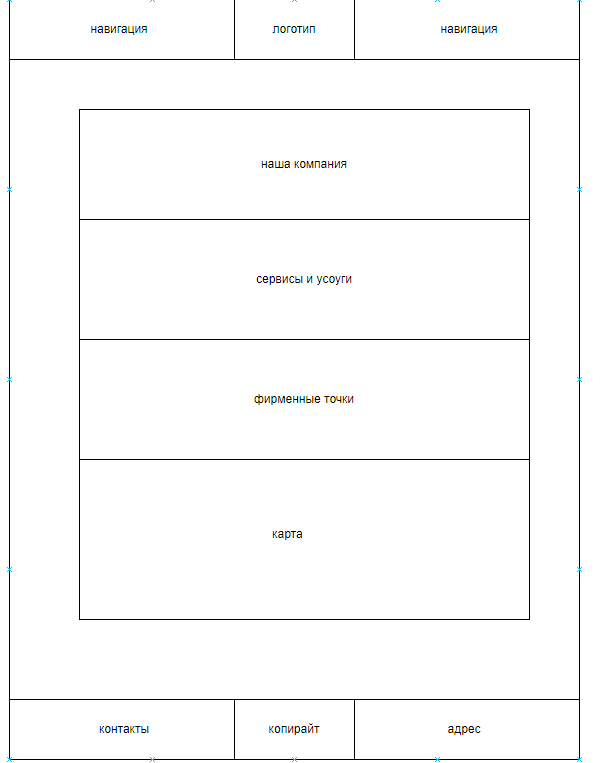


Рисунок 6 – Блочная структура страницы о нас.

Страница «Контакты» практически ничем не отличается от главной страницы. На данной страницы предоставлены все известные контакты для связи с компанией. Блочная структура страниц «Контакты» на рисунке 7.

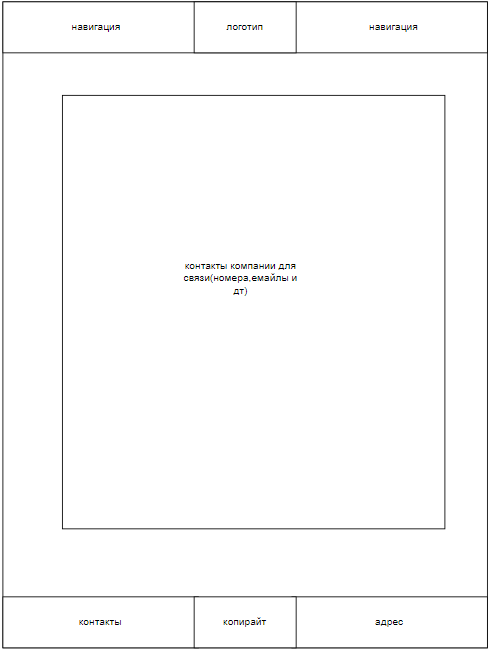


Рисунок 6 – Блочная структура страницы контакты.

3.1.2 Этап подготовки

Создание сайта - достаточно трудоемкий процесс. В его разработке принимает участие целая команда специалистов: интернет-маркетологи, дизайнеры, программисты, тестировщики и многие другие.

Разумеется, большая часть работы остается скрытой от пользователей, однако это не уменьшает ее объема. Для людей, не знакомых с данной сферой, разработка сайта может показаться достаточно легким занятием, не требующим такой большой команды и времени. Однако они сильно заблуждаются, считая процесс создания сайта легким. На деле перед получением готового продукта специалистам придется пройти множество этапов разработки сайта. В данной главе, на примере сайта компании Сам-по, разработкой которого я занимался, я рассмотрю все этапы создания сайта подробнее.

3.1.3 Этап подготовки дизайна

Создание дизайн концепции сайта – это схематически показывает, как будет выглядеть сайт. Дизайн сайта можно сравнить с эскизом, который потом будет наполнен объемом, цветом и смысловой нагрузкой.

Прототипирование – прототип страницы, который показывает вид и структуру страниц. Были созданы прототипы всех страниц сайта САМ-ПО: Главная, продукция, о нас, тесты, контакты.



Рисунок 7 – Прототип главной страницы.



Рисунок 8 – Прототип страницы продукции.



Рисунок 9 – Прототип страницы о нас.



Рисунок 10 – Прототип страницы тесты.



Рисунок 11 – Прототип страницы контакты.

3.1.4 Этап верстки сайта

На этапе дизайна все было разработано, теперь начинается этап работы с кодом. Основная задача программиста-верстальщика – перенести прототип дизайна в код, который в точности будет похож. Существует два вида программистов:

* FRONTEND
* BACKEND

На данном этапе мы остановимся на fronted, позже вернемся к backend. Frontend-программист – занимается внешней частью сайта, настраивает его внешний вид, отображение элементов и их анимацию.

Верстка – это основа сайта, которая начинается с языка разметки HTML. Он помогает программисту показать контент страницы в заданном порядке, работа с языком разметке подразумевает работу с тегами.

Основные теги, которые встречаются в работе frontend:

* <html> </html> – главный тег, в котором содержатся другие теги;
* <head> </head> – включает в себя различные SEO-элементы, код JavaScript и многое другое;
* <body> </body> – внутри этих тегов находится все содержимое страницы;
* <h1> </h1> – используется для обозначения заголовка первого уровня;
* <h2> </h2> – используется для обозначения заголовка второго уровня, после которого следуют заголовки h3, h4, h5, h6;
* <p> </p> – здесь прописывается текстовое описание, например, заголовок статьи;
* <strong> </strong> – придает тексту жирность;
* <i> </i> – текст, написанный внутри этого тега, отображается курсивом;
* <ul> </ul> – обозначает маркированный список;
* <ol> </ol> – обозначает нумерованный список;
* <li> </li> – указывает на пункты внутри списка;
* <a> </a> – позволяет добавить к тексту ссылку на другую страницу, номер телефона, почту или иной элемент с помощью атрибута href;
* <img> – используется, чтобы добавить картинку на сайт с компьютера либо из интернета;
* <table> </table> — тег для создания таблицы.

Существует два вида верстки – табличная, блочная

Блочная верстка - это отосительно условное название способов и приемов верстки, когда в большинстве веб-страниц для разметки используется CSS-свойство float, а основным строительным элементов веб-страниц является элемент <div>, то есть по сути блок. Используя свойство float и элементы div или другие элементы, можно создать структуру страницы из нескольких столбцов, как при табличной верстке, которая будет значительно гибче.

Табличная верстка — условное название метода верстки веб-документов, при котором в качестве структурной основы для расположения текстовых и графических элементов документа используются таблицы (то есть HTML-тег <table>).

В интернет портале САМ-ПО используется блочная верстка

Так же существует валидная верстка. Валидная верстка – это верстка, соответствующая стандарту W3C и означающая корректное отображение сайта на всех пользовательских устройствах. Такой подход обеспечивается строгим соблюдением правил построения кода, его оптимизации и минимизации. В результате это позволяет выводить сайт на более высокий уровень – его репутация в поисковиках сильно улучшается.

Дальше будет показан код главное страницы.

<!DOCTYPE html>

<html lang="ru">

<head>

  <meta charset="UTF-8">

  <meta http-equiv="X-UA-Compatible" content="IE=edge">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <!-- шрифты с google font-->

  <link rel="preconnect" href="https://fonts.googleapis.com">

  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

  <link

    href="https://fonts.googleapis.com/css2?family=Merriweather:wght@700&family=Raleway:wght@300;400;500;600&display=swap"

    rel="stylesheet">

  <!-- подключение стилей css-->

  <link rel="stylesheet" href="/css/style.css">

  <link rel="stylesheet" href="/css/itc-slider.css">

  <title>САМ-ПО Мороженое</title>

</head>

<body>

  <nav class="navbar">

    <div class="container\_menu">

        <ul class="nav-menu">

            <li class="nav-menu\_\_item"><a href="/product.html">Продукция</a></li>

            <li class="nav-menu\_\_item"><a href="/about us.html">О нас</a></li>

            <li class="nav-menu\_\_item logo">

                <a href="/index.html" class="logo\_\_link">

                    <img class="logo\_\_image" src="/img/Лого.png" alt="сам" class="logo">

                </a>

            </li>

            <li class="nav-menu\_\_item "><a class="submenu-link" href="">Тесты</a>

                <ul class="submenu">

                    <li><a href="/test.html">Тест на знание</a></li>

                    <li><a href="/test2.html">Тест на пригодность</a></li>

                    <li><a href="/game.html">Игра</a></li>

                </ul>

            </li>

            <li class="nav-menu\_\_item"><a href="#"> Контакты</a></li>

        </ul>

    </div>

</nav>

  <div class="container\_\_slider">

    <div class="itc-slider" data-autoplay="true" data-interval="3000" data-loop="true" data-slider="itc-slider">

      <div class="itc-slider\_\_wrapper">

        <div class="itc-slider\_\_items">

          <div class="itc-slider\_\_item">

            <div class="item\_\_text one">

              <p>Заслуженная награда - <br>

                <span class="span\_\_text"> Бренд региона</span>

              </p>

            </div>

            <img class="item\_\_img one" src="/img/Vector.png" width="560.92px" height="462.18px" alt="">

          </div>

          <div class="itc-slider\_\_item">

            <div class="item\_\_text two">

              <p>Актуально - <br>

                <span class="span\_\_text"> Бережливое производство</span>

              </p>

            </div>

            <img class="item\_\_img two" src="/img/Бережное произв.png" width="527px" height="474px" alt="">

          </div>

          <div class="itc-slider\_\_item">

            <div class="item\_\_text three">

              <p>Честный знак - <br>

                <span class="span\_\_text"> 100% гарантия качества</span>

              </p>

            </div>

            <img class="item\_\_img three" src="/img/Честный знак.png" width="526px" height="475px" alt="">

          </div>

        </div>

      </div>

      <ol class="itc-slider\_\_indicators">

        <li class="itc-slider\_\_indicator" data-slide-to="0"></li>

        <li class="itc-slider\_\_indicator" data-slide-to="1"></li>

        <li class="itc-slider\_\_indicator" data-slide-to="2"></li>

      </ol>

    </div>

  </div>>

  <div class="container\_\_icecream flex">

    <div class="icecream\_\_img">

      <img src="/img/i8h9hAKxJ\_I-transformed.png" alt="ice cream" height="900px">

    </div>

    <div class="icecream\_text">

      <h2 class="text\_\_Merriweather">Мороженое <span class="span\_\_text">САМ-ПО</span></h2>

      <p>Мороженое — это многокомпонентный молочный продукт, в состав которого входят следующие основные ингредиенты:

        цельное коровье молоко, сливки, сливочное масло, сухое

        молоко, сахар-песок, сгущённое молоко, какао-порошок,

        сироп крем-брюле. Для производства мороженого на нашем производстве используются только натуральные,

        высококачественные сырьевые компоненты. Для сложно-

        декорированного мороженого применяются различные

        фруктово-ягодные наполнители, кусочки фруктов и ягод,

        обжаренные орехи, шоколадные топпинги и мягкие

        карамели</p>

    </div>

  </div>

  <div class="container\_\_news">

    <div class="logo\_\_news">

      <h2>Новости</h2>

      <img class="img\_\_news" src="/img/Лого2.png" alt="">

    </div>

    <div class="news\_\_content">

      <div class="news\_column\_one">

        <img src="/img/news\_10\_2021 1.jpg" alt="">

        <p>Присвоение региональной символики осуществляется министерством промышленности и торговли Самарской области

          на основании решения Комиссии по присвоению региональной символики товарам, производимым на территории

          Самарской

          области. Присвоение региональной символики направлено на поддержку региональных производителей товаров.

          Не смотря на призы и награды от регионального правительства, мы всегда находимся в тесной коммуникации

          непосредственно с покупателями нашей продукции, за что получаем благодарственные письма и отзывы:</p>

        <p>Уважаемые производители, хочу сказать большое спасибо за ваш продукт. Мне понравилось мороженое Гигант

          из вафельной страны. Оно имеет приятный сливочный вкус, хрустящий рожок и покрыто моей любимой шоколадной

          глазурью. Также мороженое в меру сладкое и хорошо упаковано. <br><br>

          Вкусное мороженое поднимает настроение и помогает вырабатваться гормону счастья, а, как известно, «радостное

          сердце благотворно, как лекарство». <br><br>

          Спасибо, что продолжаете радовать нас. <br><br>

          Желаю вам всего хорошего и успехов в работе!</p>

        <p class="float">Сергей Булочкин</p>

      </div>

      <div class="news\_column\_two">

        <p>Здравствуйте, меня зовут Андрей. Мне и моей сестре Нине очень понравилось ваше мороженое «Самарское». Оно

          очень вкусное. Орехи, карамель, шоколад и мороженое идеально сочетаются и дополняют друг друга! <br><br>

          Удобная в использовании палочка, с помощью нее комфотно держать мороженое, и благодаря этому можно долго

          наслаждаться его изысканным вкусом.<br><br>

          Когда мы вкушаем ваше мороженое то, вспоминаем мудрые слова царя Соломона:»… нет лучшего для человека под

          солнцем, как есть, пить и веселиться; это сопровождает его в трудах во дни жизни его, которые дал ему Бог под

          солнцем». Это записано в Священном писании, книге Екклесиаст, глава 8, стих 15.<br><br>

          Огромное спасибо Вам за такое чудесное мороженое, которое радует нас, взрослых и детей.<br><br>

          Желаем вам и вашим семьям счастья, здоровья и благополучия!</p>

        <p class="float\_two">С уважением Андрей и Нина.</p>

        <p>Здравствуйте, уважаемый коллектив ООО «Сам-По».<br><br>

          Спасибо за изготовление очень вкусного мороженого, ванильного пломбира, в вафельном сахарном рожке

          с шоколадной глазурью.<br><br>

          Пломбир, «Гигант из вафельной страны», любимое лакомство не только детей, но и взрослых.<br><br>

          Приятно в жаркий знойный день, сидя под тенью деревьев, наслаждаться прохладой мороженого, ощущая его вкусовые

          качества.<br><br>

          Хочется поблагодарить Вас Библейскими словами царя Соломона: «Не во власти человека и то благо, чтобы есть

          и пить, и услаждать душу свою от труда своего».<br><br>

          О каком благе сказал Соломон?<br><br>

          Если вам будет интересно, с радостью поделюсь.<br><br>

          Доброго Вам здоровья и успехов!</p>

        <p>Ваш благодарный покупатель.</p>

      </div>

    </div>

  </div>

  <footer>

    <div class="container\_\_footer">

      <div class="footer\_\_items">

        <div class="footer\_\_item">

          <p class="p\_footer\_one">8 (846) 955-04-71</p>

          <p class="p\_footer\_one">secretary@sam-po.ru</p>

        </div>

        <div class="footer\_\_item">

          <p class="p\_footer\_two">© Группа компаний САМ-ПО, 2023. Все права защищены.</p>

        </div>

        <div class="footer\_\_item">

          <p class="p\_footer\_one">г. Самара, пр-д Мальцева, 9</p>

        </div>

      </div>

    </div>

  </footer>

  <script src="/js/itc-slider.js" defer></script>

</body>

</html>

Дальше идет стилизация главной страницы

\* {

    margin: 0;

    padding: 0;

    list-style: none;

    text-decoration: none;

    box-sizing: border-box;

}

body{

    background: url(../img/фон\_page-0001.jpg);

    background-repeat: no-repeat;

    background-size: cover;

}

body.two{

    background: url(../img/fon2.jpg);

    background-repeat: no-repeat;

    background-size: cover;

}

body.three{

    background: url(../img/fon3.jpg);

    background-repeat: no-repeat;

    background-size: cover;

}

body.four{

    background: url(../img/fon3.jpg);

    background-repeat: no-repeat;

    background-size: cover;

}

/\*header\*/

.container\_menu {

    background-color: #fff;

    margin: 0 auto;

    box-shadow: 0px 4px 10px rgba(0, 0, 0, 0.18);

    width: 100%;

    }

    .nav-menu {

        position: relative;

    list-style: none;

    display: flex;

    align-items: center;

    justify-content: center;

    }

    .nav-menu\_\_item {

    display: flex;

    margin: 40px 50px;

    }

    .nav-menu\_\_item a {

    font-family: 'Raleway';

    font-style: normal;

    font-weight: 500;

    font-size: 32px;

    line-height: 38px;

    text-decoration: none;

    color: #000;

    }

    .nav-menu\_\_item\_center {

    display: block;

    }

    .logo {

    text-align: center;

    width:125px;

    }

    .logo\_\_link {

    display: inline-block;

    position:absolute;

    top:18px;

    }

    .logo\_\_image {

    display: block;

    width: 140px;

    height: 140px;

    }

    .nav-menu::before {

    content: "";

    position: absolute;

    right: 865px;

    width: 9em;

    height: 3.2em;

    top: 118px;

    border-radius: 0 0 50% 50% / 0 0 100% 100%;

    background-color:#fdfdfd;

    box-shadow: 0px 4px 10px rgba(0, 0, 0, 0.18);

}

.submenu {

    background: #fff;

    position: absolute;

    left: 1;

    top: 100%;

    z-index: 5;

    width: 180px;

    opacity: 0;

    transform: scaleY(0);

    transform-origin:0 0;

    transition: .5s ease-in-out;

    -webkit-transition: .5s ease-in-out;

    -moz-transition: .5s ease-in-out;

    -ms-transition: .5s ease-in-out;

    -o-transition: .5s ease-in-out;

}

  .submenu a {

    font-family: 'Raleway';

    font-style: normal;

    font-weight: 700;

    text-align: left;

    padding: 12px 15px;

    font-size: 14px;

    border-bottom: 1px solid rgba(255,255,255,.1);

  }

  .submenu li:last-child a {border-bottom: none;}

   li:hover .submenu {

    opacity: 1;

      transform: scaleY(1);

  }

.span\_\_text{

    color: #DEBA6F;

}

.container\_\_icecream{

    width: 1588px;

    height: 884.07px;

    margin: 270px auto;

}

.flex{

    display: flex;

    justify-content:center;

    align-items: center;

}

.text\_\_Merriweather{

    font-family: 'Merriweather';

    font-style: normal;

    font-weight: 700;

    font-size: 55px;

    line-height: 69px;

    margin-bottom: 40px;

}

.icecream\_text{

    margin-left: 195px;

}

.icecream\_text p{

    font-family: 'Raleway';

    font-style: normal;

    font-weight: 400;

    font-size: 20px;

    line-height: 23px;

    width: 614px;

}

.container\_\_news{

    width: 1675px;

    height: 1873px;

    margin: 0 auto 120px;

}

.logo\_\_news{

    display: flex;

    justify-content: space-between;

    margin-bottom: 310px;

}

.img\_\_news{

    width: 328px;

    height: 326px;

    margin: 60px 250px 0 0;

}

.logo\_\_news h2{

    font-family: 'Merriweather';

    font-style: normal;

    font-weight: 700;

    font-size: 55px;

    line-height: 69px;

    margin: 230px 0 0 171px;

}

.news\_column\_one{

    width: 635px;

}

.news\_column\_one img{

    margin-bottom: 37px;

}

.news\_column\_one p{

    font-family: 'Raleway';

    font-style: normal;

    font-weight: 400;

    font-size: 20px;

    line-height: 23px;

    margin-bottom: 37px;

}

.float{

    float: right;

}

.news\_\_content{

    display: flex;

    justify-content:space-between;

}

.news\_column\_two{

    width: 675px;

    justify-content: flex-end;

}

.news\_column\_two p{

    font-family: 'Raleway';

    font-style: normal;

    font-weight: 400;

    font-size: 20px;

    line-height: 23px;

}

.float\_two{

    text-align: right;

    margin-top: 55px;

    margin-bottom: 45px;

}

.news\_column\_two p:last-child  {

    text-align: right;

    margin-top: 76px;

}

footer{

    background-color: #fff;

}

.container\_\_footer{

    height: 138px;

    box-shadow: 0px 4px 10px rgba(0, 0, 0, 0.18);

}

.footer\_\_item{

    width: 33%;

    display: inline-block;

    text-align: center;

    padding-top: 50px;

}

.p\_footer\_one{

    font-family: 'Raleway';

    font-style: normal;

    font-weight: 400;

    font-size: 20px;

    line-height: 23px;

}

.p\_footer\_two{

    font-family: 'Raleway';

    font-style: normal;

    font-weight: 400;

    font-size: 15px;

    line-height: 18px;

}

Почему не имеет смысла показывать остальные страницы? Из-за того, что все страницы практически похожи, меняется только контент страниц. В других главах буден показан код только скриптов, которые были разработаны для сайта

3.1.5 Этап написание скриптов для сайта

Скрипт - это набор команд, которые выполняют конкретную задачу. Скрипты могут быть маленькими и отвечать за запуск каких-то простых служб операционной системы, так и большими, сравнивая переменные и выводя результат на сайте.

В этом разделе мы поговорим о всех видах скриптах, которые были созданы для сайта САМ-ПО

На главной странице есть слайдер, он написан на java script, скрипт отвечает за автоматическое переключение слайдера, так же можно настроить плавность и скорость переключателя через этот скрипт.

class ItcSlider {

  static #EL\_WRAPPER = 'wrapper';

  static #EL\_ITEMS = 'items';

  static #EL\_ITEM = 'item';

  static #EL\_ITEM\_ACTIVE = 'item\_active';

  static #EL\_INDICATOR = 'indicator';

  static #EL\_INDICATOR\_ACTIVE = 'indicator\_active';

  static #BTN\_PREV = 'btn\_prev';

  static #BTN\_NEXT = 'btn\_next';

  static #BTN\_HIDE = 'btn\_hide';

  static #TRANSITION\_NONE = 'transition-none';

  static #instances = [];

  #config;

  #state;

  constructor(el, config = {}, prefix = 'itc-slider\_\_') {

    this.#state = {

      prefix, // префикс для классов

      el, // элемент который нужно активировать как ItcSlider

      elWrapper: el.querySelector(`.${prefix}${this.constructor.#EL\_WRAPPER}`), // элемент с #CLASS\_WRAPPER

      elItems: el.querySelector(`.${prefix}${this.constructor.#EL\_ITEMS}`), // элемент, в котором находятся слайды

      elListItem: el.querySelectorAll(`.${prefix}${this.constructor.#EL\_ITEM}`), // список элементов, являющиеся слайдами

      btnPrev: el.querySelector(`.${prefix}${this.constructor.#BTN\_PREV}`), // кнопка, для перехода к предыдущему слайду

      btnNext: el.querySelector(`.${prefix}${this.constructor.#BTN\_NEXT}`), // кнопка, для перехода к следующему слайду

      btnClassHide: prefix + this.constructor.#BTN\_HIDE, // класс для скрытия кнопки

      exOrderMin: 0,

      exOrderMax: 0,

      exItemMin: null,

      exItemMax: null,

      exTranslateMin: 0,

      exTranslateMax: 0,

      direction: 'next', // направление смены слайдов

      intervalId: null, // id таймера

      isSwiping: false,

      swipeX: 0,

    };

    this.#config = {

      loop: true, autoplay: false, interval: 5000, refresh: true, swipe: true, ...config

    };

    this.#init();

    this.#attachEvents();

  }

  static getInstance(elSlider) {

    const found = this.#instances.find((el) => el.target === elSlider);

    if (found) {

      return found.instance;

    }

    return null;

  }

  /\*\*

   \* @param {String|HTMLElement} target

   \* @param {Object} config

   \* @param {String} prefix

   \*/

  static getOrCreateInstance(target, config = {}, prefix = 'itc-slider\_\_') {

    try {

      const elSlider = typeof target === 'string' ? document.querySelector(target) : target;

      const result = this.getInstance(elSlider);

      if (result) {

        return result;

      }

      const slider = new this(elSlider, config, prefix);

      this.#instances.push({ target: elSlider, instance: slider });

      return slider;

    } catch (e) {

      console.error(e);

    }

  }

  // статический метод для активирования элементов как ItcSlider на основе data-атрибутов

  static createInstances() {

    document.querySelectorAll('[data-slider="itc-slider"]').forEach((el) => {

      const { dataset } = el;

      const params = {};

      Object.keys(dataset).forEach((key) => {

        if (key === 'slider') {

          return;

        }

        let value = dataset[key];

        value = value === 'true' ? true : value;

        value = value === 'false' ? false : value;

        value = Number.isNaN(Number(value)) ? Number(value) : value;

        params[key] = value;

      });

      this.getOrCreateInstance(el, params);

    });

  }

  next() {

    this.#state.direction = 'next';

    this.#move();

  }

  prev() {

    this.#state.direction = 'prev';

    this.#move();

  }

  moveTo(index) {

    this.#moveTo(index);

  }

  reset() {

    this.#reset();

  }

  dispose() {

    this.#detachEvents();

    const transitionNoneClass = this.#state.prefix + this.constructor.#TRANSITION\_NONE;

    const activeClass = this.#state.prefix + this.constructor.#EL\_ITEM\_ACTIVE;

    this.#autoplay('stop');

    this.#state.elItems.classList.add(transitionNoneClass);

    this.#state.elItems.style.transform = '';

    this.#state.elListItem.forEach((el) => {

      el.style.transform = '';

      el.classList.remove(activeClass);

    });

    const selIndicators = `${this.#state.prefix}${this.constructor.#EL\_INDICATOR\_ACTIVE}`;

    document.querySelectorAll(`.${selIndicators}`).forEach((el) => {

      el.classList.remove(selIndicators);

    });

    this.#state.elItems.offsetHeight;

    this.#state.elItems.classList.remove(transitionNoneClass);

    const index = this.constructor.#instances.findIndex((el) => el.target === this.#state.el);

    this.constructor.#instances.splice(index, 1);

  }

  #onClick(e) {

    if (!(e.target.closest('.itc-slider\_\_btn') || e.target.closest('.itc-slider\_\_indicators'))) {

      return;

    }

    e.preventDefault();

    const classBtnPrev = this.#state.prefix + this.constructor.#BTN\_PREV;

    const classBtnNext = this.#state.prefix + this.constructor.#BTN\_NEXT;

    this.#autoplay('stop');

    if (e.target.closest(`.${classBtnPrev}`) || e.target.closest(`.${classBtnNext}`)) {

      this.#state.direction = e.target.closest(`.${classBtnPrev}`) ? 'prev' : 'next';

      this.#move();

    } else if (e.target.dataset.slideTo) {

      const index = parseInt(e.target.dataset.slideTo, 10);

      this.#moveTo(index);

    }

    this.#config.loop ? this.#autoplay() : null;

  }

  #onMouseEnter() {

    this.#autoplay('stop');

  }

  #onMouseLeave() {

    this.#autoplay();

  }

  #onResize() {

    window.requestAnimationFrame(this.#reset.bind(this));

  }

  #onSwipeStart(e) {

    this.#autoplay('stop');

    const event = e.type.search('touch') === 0 ? e.touches[0] : e;

    this.#state.swipeX = event.clientX;

    this.#state.isSwiping = true;

  }

  #onSwipeEnd(e) {

    if (!this.#state.isSwiping) {

      return;

    }

    const event = e.type.search('touch') === 0 ? e.changedTouches[0] : e;

    const diffPos = this.#state.swipeX - event.clientX;

    if (diffPos > 50) {

      this.#state.direction = 'next';

      this.#move();

    } else if (diffPos < -50) {

      this.#state.direction = 'prev';

      this.#move();

    }

    this.#state.isSwiping = false;

    if (this.#config.loop) {

      this.#autoplay();

    }

  }

  #onTransitionStart() {

    if (this.#state.isBalancing) {

      return;

    }

    this.#state.isBalancing = true;

    window.requestAnimationFrame(this.#balanceItems.bind(this));

  }

  #onTransitionEnd() {

    this.#state.isBalancing = false;

  }

  #onDragStart(e) {

    e.preventDefault();

  }

  #onVisibilityChange() {

    if (document.visibilityState === 'hidden') {

      this.#autoplay('stop');

    } else if (document.visibilityState === 'visible' && this.#config.loop) {

      this.#autoplay();

    }

  }

  #attachEvents() {

    this.#state.events = {

      click: [this.#state.el, this.#onClick.bind(this), true],

      mouseenter: [this.#state.el, this.#onMouseEnter.bind(this), true],

      mouseleave: [this.#state.el, this.#onMouseLeave.bind(this), true],

      resize: [window, this.#onResize.bind(this), this.#config.refresh],

      'itc-slider\_\_transition-start': [this.#state.elItems, this.#onTransitionStart.bind(this), this.#config.loop],

      transitionend: [this.#state.elItems, this.#onTransitionEnd.bind(this), this.#config.loop],

      touchstart: [this.#state.el, this.#onSwipeStart.bind(this), this.#config.swipe],

      mousedown: [this.#state.el, this.#onSwipeStart.bind(this), this.#config.swipe],

      touchend: [document, this.#onSwipeEnd.bind(this), this.#config.swipe],

      mouseup: [document, this.#onSwipeEnd.bind(this), this.#config.swipe],

      dragstart: [this.#state.el, this.#onDragStart.bind(this), true],

      visibilitychange: [document, this.#onVisibilityChange.bind(this), true]

    };

    Object.keys(this.#state.events).forEach((type) => {

      if (this.#state.events[type][2]) {

        const el = this.#state.events[type][0];

        const fn = this.#state.events[type][1];

        el.addEventListener(type, fn);

      }

    });

  }

  #detachEvents() {

    Object.keys(this.#state.events).forEach((type) => {

      if (this.#state.events[type][2]) {

        const el = this.#state.events[type][0];

        const fn = this.#state.events[type][1];

        el.removeEventListener(type, fn);

      }

    });

  }

  #autoplay(action) {

    if (!this.#config.autoplay) {

      return;

    }

    if (action === 'stop') {

      clearInterval(this.#state.intervalId);

      this.#state.intervalId = null;

      return;

    }

    if (this.#state.intervalId === null) {

      this.#state.intervalId = setInterval(() => {

        this.#state.direction = 'next';

        this.#move();

      }, this.#config.interval);

    }

  }

  #balanceItems() {

    if (!this.#state.isBalancing) {

      return;

    }

    const wrapperRect = this.#state.elWrapper.getBoundingClientRect();

    const targetWidth = wrapperRect.width / this.#state.countActiveItems / 2;

    const countItems = this.#state.elListItem.length;

    if (this.#state.direction === 'next') {

      const exItemRectRight = this.#state.exItemMin.getBoundingClientRect().right;

      if (exItemRectRight < wrapperRect.left - targetWidth) {

        const elFound = this.#state.els.find((item) => item.el === this.#state.exItemMin);

        elFound.order = this.#state.exOrderMin + countItems;

        const translate = this.#state.exTranslateMin + countItems \* this.#state.width;

        elFound.translate = translate;

        this.#state.exItemMin.style.transform = `translate3D(${translate}px, 0px, 0.1px)`;

        this.#updateExProperties();

      }

    } else {

      const exItemRectLeft = this.#state.exItemMax.getBoundingClientRect().left;

      if (exItemRectLeft > wrapperRect.right + targetWidth) {

        const elFound = this.#state.els.find((item) => item.el === this.#state.exItemMax);

        elFound.order = this.#state.exOrderMax - countItems;

        const translate = this.#state.exTranslateMax - countItems \* this.#state.width;

        elFound.translate = translate;

        this.#state.exItemMax.style.transform = `translate3D(${translate}px, 0px, 0.1px)`;

        this.#updateExProperties();

      }

    }

    window.requestAnimationFrame(this.#balanceItems.bind(this));

  }

  #updateClasses() {

    const activeClass = this.#state.prefix + this.constructor.#EL\_ITEM\_ACTIVE;

    this.#state.activeItems.forEach((item, index) => {

      if (item) {

        this.#state.elListItem[index].classList.add(activeClass);

      } else {

        this.#state.elListItem[index].classList.remove(activeClass);

      }

      const elListIndicators = this.#state.el.querySelectorAll(`.${this.#state.prefix}${this.constructor.#EL\_INDICATOR}`);

      if (elListIndicators.length && item) {

        elListIndicators[index].classList.add(`${this.#state.prefix}${this.constructor.#EL\_INDICATOR\_ACTIVE}`);

      } else if (elListIndicators.length && !item) {

        elListIndicators[index].classList.remove(`${this.#state.prefix}${this.constructor.#EL\_INDICATOR\_ACTIVE}`);

      }

    });

  }

  #move() {

    const widthItem = this.#state.direction === 'next' ? -this.#state.width : this.#state.width;

    const transform = this.#state.translate + widthItem;

    if (!this.#config.loop) {

      const limit = this.#state.width \* (this.#state.elListItem.length - this.#state.countActiveItems);

      if (transform < -limit || transform > 0) {

        return;

      }

      if (this.#state.btnPrev) {

        this.#state.btnPrev.classList.remove(this.#state.btnClassHide);

        this.#state.btnNext.classList.remove(this.#state.btnClassHide);

      }

      if (this.#state.btnPrev && transform === -limit) {

        this.#state.btnNext.classList.add(this.#state.btnClassHide);

      } else if (this.#state.btnPrev && transform === 0) {

        this.#state.btnPrev.classList.add(this.#state.btnClassHide);

      }

    }

    if (this.#state.direction === 'next') {

      this.#state.activeItems = [...this.#state.activeItems.slice(-1), ...this.#state.activeItems.slice(0, -1)];

    } else {

      this.#state.activeItems = [...this.#state.activeItems.slice(1), ...this.#state.activeItems.slice(0, 1)];

    }

    this.#updateClasses();

    this.#state.translate = transform;

    this.#state.elItems.style.transform = `translate3D(${transform}px, 0px, 0.1px)`;

    this.#state.elItems.dispatchEvent(new CustomEvent('itc-slider\_\_transition-start', {

      bubbles: true

    }));

  }

  #moveTo(index) {

    const delta = this.#state.activeItems.reduce((acc, current, currentIndex) => {

      const diff = current ? index - currentIndex : acc;

      return Math.abs(diff) < Math.abs(acc) ? diff : acc;

    }, this.#state.activeItems.length);

    if (delta !== 0) {

      this.#state.direction = delta > 0 ? 'next' : 'prev';

      for (let i = 0; i < Math.abs(delta); i++) {

        this.#move();

      }

    }

  }

  // приватный метод для выполнения первичной иницианализации

  #init() {

    // состояние элементов

    this.#state.els = [];

    // текущее значение translate

    this.#state.translate = 0;

    // позиции активных элементов

    this.#state.activeItems = [];

    // состояние элементов

    this.#state.isBalancing = false;

    // получаем gap между слайдами

    const gap = parseFloat(getComputedStyle(this.#state.elItems).gap) || 0;

    // ширина одного слайда

    this.#state.width = this.#state.elListItem[0].getBoundingClientRect().width + gap;

    // ширина #EL\_WRAPPER

    const widthWrapper = this.#state.elWrapper.getBoundingClientRect().width;

    // количество активных элементов

    this.#state.countActiveItems = Math.round(widthWrapper / this.#state.width);

    this.#state.elListItem.forEach((el, index) => {

      el.style.transform = '';

      this.#state.activeItems.push(index < this.#state.countActiveItems ? 1 : 0);

      this.#state.els.push({

        el, index, order: index, translate: 0

      });

    });

    if (this.#config.loop) {

      const lastIndex = this.#state.elListItem.length - 1;

      const translate = -(lastIndex + 1) \* this.#state.width;

      this.#state.elListItem[lastIndex].style.transform = `translate3D(${translate}px, 0px, 0.1px)`;

      this.#state.els[lastIndex].order = -1;

      this.#state.els[lastIndex].translate = translate;

      this.#updateExProperties();

    } else if (this.#state.btnPrev) {

      this.#state.btnPrev.classList.add(this.#state.btnClassHide);

    }

    this.#updateClasses();

    this.#autoplay();

  }

  #reset() {

    const transitionNoneClass = this.#state.prefix + this.constructor.#TRANSITION\_NONE;

    // получаем gap между слайдами

    const gap = parseFloat(getComputedStyle(this.#state.elItems).gap) || 0;

    // ширина одного слайда

    const widthItem = this.#state.elListItem[0].getBoundingClientRect().width + gap;

    const widthWrapper = this.#state.elWrapper.getBoundingClientRect().width;

    const countActiveEls = Math.round(widthWrapper / widthItem);

    if (widthItem === this.#state.width && countActiveEls === this.#state.countActiveItems) {

      return;

    }

    this.#autoplay('stop');

    this.#state.elItems.classList.add(transitionNoneClass);

    this.#state.elItems.style.transform = 'translate3D(0px, 0px, 0.1px)';

    this.#init();

    window.requestAnimationFrame(() => {

      this.#state.elItems.classList.remove(transitionNoneClass);

    });

  }

  #updateExProperties() {

    const els = this.#state.els.map((item) => item.el);

    const orders = this.#state.els.map((item) => item.order);

    this.#state.exOrderMin = Math.min(...orders);

    this.#state.exOrderMax = Math.max(...orders);

    const min = orders.indexOf(this.#state.exOrderMin);

    const max = orders.indexOf(this.#state.exOrderMax);

    this.#state.exItemMin = els[min];

    this.#state.exItemMax = els[max];

    this.#state.exTranslateMin = this.#state.els[min].translate;

    this.#state.exTranslateMax = this.#state.els[max].translate;

  }

}

ItcSlider.createInstances();

Дальше идет скрипт для тестов  
const questions = [

    {

        question: "В какой день отмечают международный день эскимо? ?",

        optionA: "24 декабря",

        optionB: "24 января",

        optionC: "24 февраля",

        optionD: "24 сентября",

        correctOption: "optionB"

    },

    {

        question: "С какими добавками ел мороженое Кевин Маккаллистер в фильме «Один дома»?",

        optionA: "С коктейльной вишней, зефиром и шоколадным соусом",

        optionB: "С бананом, клубникой и орехами",

        optionC: "С ананасом и шоколадом",

        optionD: "Без дополнительных ингредиентов",

        correctOption: "optionA"

    },

    {

        question: "Как в Японии называют десерт из стружки льда с сиропом?",

        optionA: "Моти",

        optionB: "Сэмбэй",

        optionC: "Какигори",

        optionD: "Мамбао",

        correctOption: "optionC"

    },

    {

        question: "Сколько стоил вафельный рожок в «Макдоналдсе» в Москве десять лет назад?",

        optionA: "13 рублей",

        optionB: "14 рублей",

        optionC: "15 рублей",

        optionD: "16 рублей",

        correctOption: "optionC"

    },

    {

        question: "Какой российской марки мороженого не существует?" ,

        optionA: "«Миссис Айси»",

        optionB: "«Ледяная долина»",

        optionC: "«Отмороженное»",

        optionD: "Гигант",

        correctOption: "optionB"

    },

    {

        question: "Как называют кофейный напиток с шариком ванильного мороженого?",

        optionA: "Флэт-уайт",

        optionB: "Глясе",

        optionC: "Раф",

        optionD: "Бамбл",

        correctOption: "optionB"

    },

    {

        question: "Существует ли формула поедания мороженого?",

        optionA: "Да",

        optionB: "Конечно, нет. Дайте поесть спокойно!",

        optionC: "Я сам её придумал",

        optionD: "Нет,но возможно и да",

        correctOption: "optionA"

    },

    {

        question: "Какую двухметровую фигуру изготовили из мороженого в день нового тысячелетия 31.12.2000 в Москве?",

        optionA: "Снеговик",

        optionB: "Снегурочка",

        optionC: "Кролик",

        optionD: "Медведь",

        correctOption: "optionA"

    },

    {

        question: "Каких вкусов мороженого не существует?",

        optionA: "С гречкой, со свеклой",

        optionB: "С чернилами каракатицы, с вишневым пивом или с белыми грибами",

        optionC: "С бататом",

        optionD: "Все вышеперечисленные ингредиенты могут быть основой мороженого",

        correctOption: "optionD"

    },

    {

        question: "В какой стране едят замороженный десерт кульфи?",

        optionA: "Иран",

        optionB: "Египет",

        optionC: "Индия",

        optionD: "Япония",

        correctOption: "optionC"

    }

]

let shuffledQuestions = []

function handleQuestions() {

    while (shuffledQuestions.length <= 9) {

        const random = questions[Math.floor(Math.random() \* questions.length)]

        if (!shuffledQuestions.includes(random)) {

            shuffledQuestions.push(random)

        }

    }

}

let questionNumber = 1

let playerScore = 0

let wrongAttempt = 0

let indexNumber = 0

function NextQuestion(index) {

    handleQuestions()

    const currentQuestion = shuffledQuestions[index]

    document.getElementById("question-number").innerHTML = questionNumber

    document.getElementById("player-score").innerHTML = playerScore

    document.getElementById("display-question").innerHTML = currentQuestion.question;

    document.getElementById("option-one-label").innerHTML = currentQuestion.optionA;

    document.getElementById("option-two-label").innerHTML = currentQuestion.optionB;

    document.getElementById("option-three-label").innerHTML = currentQuestion.optionC;

    document.getElementById("option-four-label").innerHTML = currentQuestion.optionD;

}

function checkForAnswer() {

    const currentQuestion = shuffledQuestions[indexNumber]

    const currentQuestionAnswer = currentQuestion.correctOption

    const options = document.getElementsByName("option");

    let correctOption = null

    options.forEach((option) => {

        if (option.value === currentQuestionAnswer) {

            correctOption = option.labels[0].id

        }

    })

    if (options[0].checked === false && options[1].checked === false && options[2].checked === false && options[3].checked == false) {

        document.getElementById('option-modal').style.display = "flex"

    }

    options.forEach((option) => {

        if (option.checked === true && option.value === currentQuestionAnswer) {

            document.getElementById(correctOption).style.backgroundColor = "green"

            playerScore++

            indexNumber++

            setTimeout(() => {

                questionNumber++

            }, 1000)

        }

        else if (option.checked && option.value !== currentQuestionAnswer) {

            const wrongLabelId = option.labels[0].id

            document.getElementById(wrongLabelId).style.backgroundColor = "red"

            document.getElementById(correctOption).style.backgroundColor = "green"

            wrongAttempt++

            indexNumber++

            setTimeout(() => {

                questionNumber++

            }, 1000)

        }

    })

}

function handleNextQuestion() {

    checkForAnswer()

    unCheckRadioButtons()

    setTimeout(() => {

        if (indexNumber <= 9) {

            NextQuestion(indexNumber)

        }

        else {

            handleEndGame()

        }

        resetOptionBackground()

    }, 1000);

}

function resetOptionBackground() {

    const options = document.getElementsByName("option");

    options.forEach((option) => {

        document.getElementById(option.labels[0].id).style.backgroundColor = ""

    })

}

function unCheckRadioButtons() {

    const options = document.getElementsByName("option");

    for (let i = 0; i < options.length; i++) {

        options[i].checked = false;

    }

}

function handleEndGame() {

    let remark = null

    let remarkColor = null

    if (playerScore <= 3) {

        remark = "Плохие оценки, продолжай практиковаться."

        remarkColor = "red"

    }

    else if (playerScore >= 4 && playerScore < 7) {

        remark = "Средние оценки, ты можешь добиться большего."

        remarkColor = "orange"

    }

    else if (playerScore >= 7) {

        remark = "Отлично, продолжайте в том же духе."

        remarkColor = "green"

    }

    const playerGrade = (playerScore / 10) \* 100

    document.getElementById('remarks').innerHTML = remark

    document.getElementById('remarks').style.color = remarkColor

    document.getElementById('grade-percentage').innerHTML = playerGrade

    document.getElementById('wrong-answers').innerHTML = wrongAttempt

    document.getElementById('right-answers').innerHTML = playerScore

    document.getElementById('score-modal').style.display = "flex"

}

function closeScoreModal() {

    questionNumber = 1

    playerScore = 0

    wrongAttempt = 0

    indexNumber = 0

    shuffledQuestions = []

    NextQuestion(indexNumber)

    document.getElementById('score-modal').style.display = "none"

}

function closeOptionModal() {

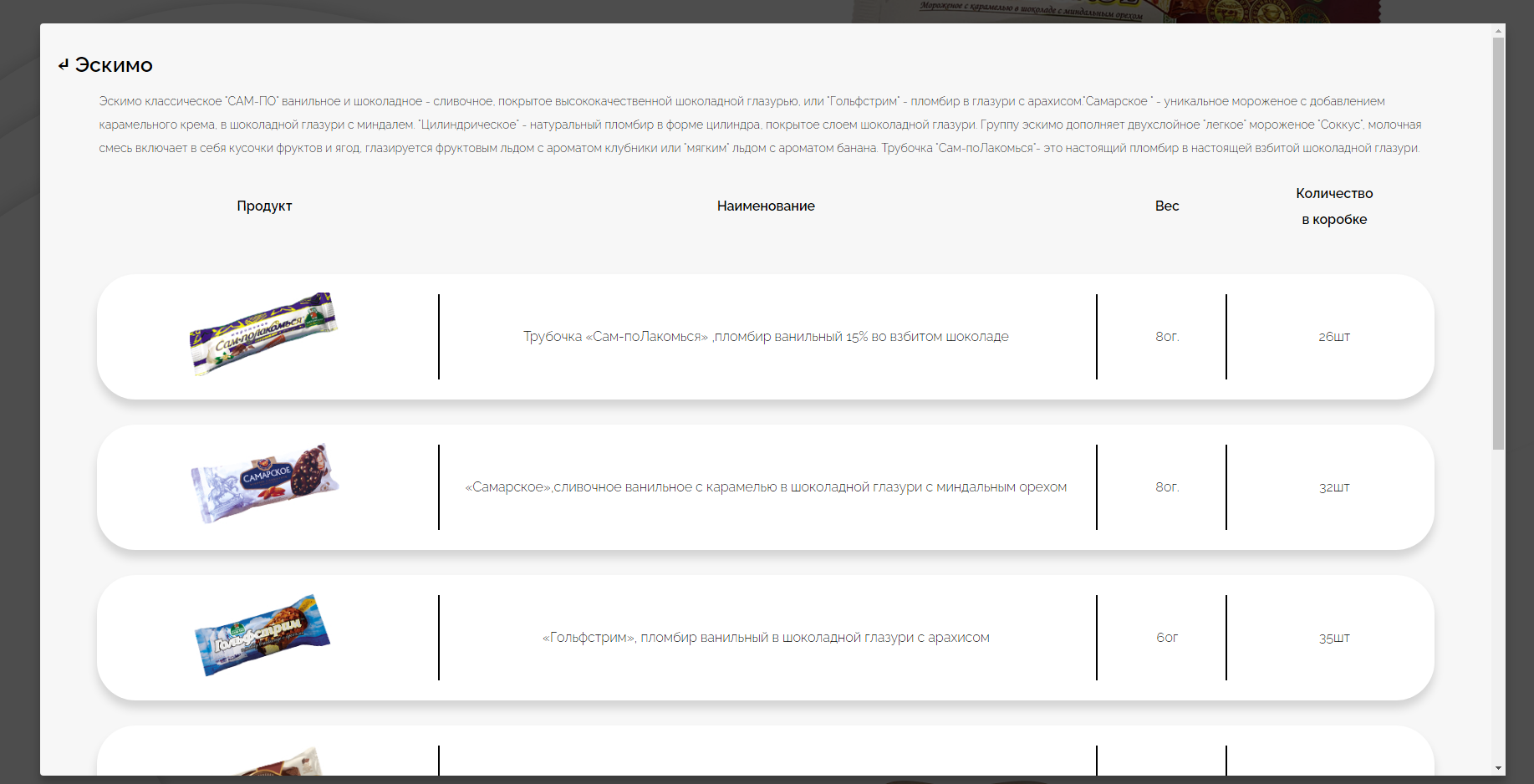
    document.getElementById('option-modal').style.display = "none"

}

3.1.6 Создание модальных окон

**Модальное** **окно́** в графическом интерфейсе пользователя — **окно**, которое блокирует работу пользователя с родительским приложением до тех пор, пока пользователь — это **окно** не закроет. **Модальными** преимущественно реализованы диалоговые **окна**. Также **модальные** **окна** часто используются для привлечения внимания пользователя к важному событию или критической ситуации.

На вкладке продукции реализуются модальные окна. Их реализация, представленная ниже:



<div class="popup-fade">

    <div class="popup">

      <a class="popup-close" href="#">↵ Эскимо</a>

      <p>Эскимо классическое "САМ-ПО" ванильное и шоколадное - сливочное, покрытое высококачественной шоколадной

        глазурью, или "Гольфстрим" - пломбир в глазури с арахисом."Самарское " - уникальное мороженое с добавлением

        карамельного крема, в шоколадной глазури с миндалем. "Цилиндрическое" - натуральный пломбир в форме цилиндра,

        покрытое слоем шоколадной глазури. Группу эскимо дополняет двухслойное "легкое" мороженое "Соккус", молочная

        смесь включает в себя кусочки фруктов и ягод, глазируется фруктовым льдом с ароматом клубники или "мягким" льдом

        с ароматом банана. Трубочка "Сам-поЛакомься"- это настоящий пломбир в настоящей взбитой шоколадной глазури.</p>

      <div class="popup-info">

      </div>

      <div class="divTable">

        <div class="divTableBody">

          <div class="divTableRow topmodal">

            <div class="divTableCell headermodal width25">Продукт</div>

            <div class="divTableCell headermodal width50 ">Наименование</div>

            <div class="divTableCell headermodal width10">Вес</div>

            <div class="divTableCell headermodal width15">Количество <br>

              в коробке</div>

          </div>

          <div class="divTableRow backmodal">

            <div class="divTableCell headermodal2 width25"><img class="img-modal" src="../img/modal\_img/1.jpg" alt="">

            </div>

            <div class="divTableCell headermodal2 width50 beforeline">Трубочка «Сам-поЛакомься» ,пломбир ванильный 15%

              во взбитом шоколаде</div>

            <div class="divTableCell headermodal2 width10 beforeline10">80г.</div>

            <div class="divTableCell headermodal2 width15">26шт <br>

            </div>

          </div>

          <div class="divTableRow backmodal">

            <div class="divTableCell headermodal2 width25"><img class="img-modal" src="../img/modal\_img/2.jpg" alt="">

            </div>

            <div class="divTableCell headermodal2 width50 beforeline">«Самарское»,сливочное ванильное с карамелью в

              шоколадной глазури с миндальным орехом</div>

            <div class="divTableCell headermodal2 width10 beforeline10">80г.</div>

            <div class="divTableCell headermodal2 width15">32шт

            </div>

          </div>

          <div class="divTableRow backmodal">

            <div class="divTableCell headermodal2 width25"><img class="img-modal" src="../img/modal\_img/3.jpg" alt="">

            </div>

            <div class="divTableCell headermodal2 width50 beforeline">«Гольфстрим», пломбир ванильный в шоколадной

              глазури с арахисом</div>

            <div class="divTableCell headermodal2 width10 beforeline10">60г</div>

            <div class="divTableCell headermodal2 width15">35шт

            </div>

          </div>

          <div class="divTableRow backmodal">

            <div class="divTableCell headermodal2 width25"><img class="img-modal" src="../img/modal\_img/4.jpg" alt="">

            </div>

            <div class="divTableCell headermodal2 width50 beforeline">«Эскимо Два шоколада» пломбир шоколадный 12% в

              шоколадной глазури</div>

            <div class="divTableCell headermodal2 width10 beforeline10">75г</div>

            <div class="divTableCell headermodal2 width15">35шт

            </div>

          </div>

          <div class="divTableRow backmodal">

            <div class="divTableCell headermodal2 width25"><img class="img-modal" src="../img/modal\_img/5.jpg" alt="">

            </div>

            <div class="divTableCell headermodal2 width50 beforeline">«Пломбир» Цилиндрическое, пломбир ванильный в

              шоколадной глазури</div>

            <div class="divTableCell headermodal2 width10 beforeline10">63г</div>

            <div class="divTableCell headermodal2 width15">17шт

            </div>

          </div>

          <div class="divTableRow backmodal">

            <div class="divTableCell headermodal2 width25"><img class="img-modal" src="../img/modal\_img/6.jpg" alt="">

            </div>

            <div class="divTableCell headermodal2 width50 beforeline">«Соккус» Клубника, двухслойное мороженое с

              кусочками клубники, покрытое фруктовым льдом</div>

            <div class="divTableCell headermodal2 width10 beforeline10">70г.</div>

            <div class="divTableCell headermodal2 width15">32шт

            </div>

          </div>

          <div class="divTableRow backmodal">

            <div class="divTableCell headermodal2 width25"><img class="img-modal" src="../img/modal\_img/7.jpg" alt="">

            </div>

            <div class="divTableCell headermodal2 width50 beforeline">«Соккус» Банан, двухслойное мороженое с кусочками

              банана,

              покрытое мягким фруктовым льдом</div>

            <div class="divTableCell headermodal2 width10 beforeline10">70г.</div>

            <div class="divTableCell headermodal2 width15">32шт

            </div>

          </div>

        </div>

      </div>

    </div>

  </div>

$(document).ready(function($) {

    $('.popup-open').click(function() {

        $('.popup-fade').fadeIn();

        return false;

    });

    $('.popup-close').click(function() {

        $(this).parents('.popup-fade').fadeOut();

        return false;

    });

    $(document).keydown(function(e) {

        if (e.keyCode === 27) {

            e.stopPropagation();

            $('.popup-fade').fadeOut();

        }

    });

    $('.popup-fade').click(function(e) {

        if ($(e.target).closest('.popup').length == 0) {

            $(this).fadeOut();

        }

    });

});

## 3.2 Технологии, использованные для разработки сайта

Любой web-сайт состоит из двух разных частей: визуальный и программный. Визуальная часть отвечает за то, что пользователь может взаимодействовать с формами, разделами, картинками, кнопками, а программная часть отвечает за действия пользователя, то есть обрабатывает информацию, которую делает человек.

Frontend использует такие технологии, как HTML, CSS, JS. Именно на этих трех технологиях строиться визуальная часть сайта. В backend ситуация похожая, но все же имеет различия в технологиях. На сегодняшний день 80% сайтов используют в backend язык программирования PHP. На основе этого есть 2 основных варианта разработки. Первый вариант это – CMS, второй – PHP Фреймворки.

CMS — это готовые системы управления сайтом. В большинстве своем это бесплатные платформы.

PHP Фреймворки дороже, сложнее и у них больший срок разработки. И если для создания сайта, на WordPress, не надо особо уметь программировать, то, чтобы сделать сайт на Фреймворке PHP, надо иметь хорошее понимание кода.

# ЗАКЛЮЧЕНИЕ

# СПИСОК ЛИТЕРАТУРЫ