**АЗЕРБАЙДЖАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ НЕФТИ И ПРОМЫШЛЕННОСТИ**

**Факультет: Информационные технологии и управление**

**Кафедра: Компьютерная инженерия**

**Предмет: «Веб-системы и технологии»**

Курсовая работа

**Тема** «Разработка онлайн информационной системы для продуктовых магазинов»

**Группа:** 680.22

**Курc:** 3

**Специальность: 050616** Информационные технологии

**Студент**: Алимов Зия

**Руководитель:** **асс. Халилов М. Э.**

**Зав.кафедрой: доц.Рагимова Н.А.**

**Баку – 2023**

**АЗЕРБАЙДЖАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ НЕФТИ И ПРОМЫШЛЕННОСТИ**

**АЗЕРБАЙДЖАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ НЕФТИ И ПРОМЫШЛЕННОСТИ**

**ЗАДАНИЕ ПО КУРСОВОЙ РАБОТЕ**

**ФАКУЛЬТЕТ**: Информационные технологии и управление

**КАФЕДРА**: Компьютерная инженерия

**Группа** 680.22 **курс** III

**Специальность** **050616** Информационные технологии

**Студент** Алимов Зия

**Зачетная книжка**

**Руководитель курсовой работы**  **асс. Халилов М. Э.**

**Срок выдачи**

**Дата сдачи**

**Тема курсовой работы** «Разработка онлайн информационной системы

для продуктовых магазинов»

**Отзыв руководителя курсовой**:

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

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

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

**Подпись студента**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Подпись рук.курсовой**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Подпись зав.кафедрой**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Дата защиты курсовой**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**Оценка**\_\_\_\_\_\_\_\_\_\_

**Пред.комиссии**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

**Члены комиссии**: 1.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

2.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

3.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

**СОДЕРЖАНИЕ**

**ВВЕДЕНИЕ………………………………………………………………………4**

1. **ТЕОРЕТИЧЕСКАЯ ЧАСТЬ………...………………………………………6**
   1. **Web-программирование…………………………………………….……...6**
   2. **HTML и XHTML………………………………………………...…………..7**
   3. **Dynamic HTML……………………………………………………………..13**

**1.4.1 Каскадные таблицы стилей……………………………………...13**

**1.4.2 JavaScript…………………………………………………………...16**

**1.5 Bootstrap……………………………………………………………………..17**

1. **ПРАКТИЧЕСКАЯ ЧАСТЬ…………………………………………...……20**
   1. **Описание предметной области…………………………………………...20**
   2. **Создание окружения для проекта………………………………………..21**
   3. **Верстка HTML страницы…………………………………………………21**
   4. **Использование стилей CSS………………………………………………..29**

**3.РЕЗУЛЬТАТ…………………………………………………………………..44 4.ЛИТЕРАТУРА………………………………………………………………..45**

**5.ПРИЛОЖЕНИЯ……………………………………………………………...46**

**ВВЕДЕНИЕ**

Веб-системы и технологии охватывают разнообразные языки программирования, инструменты, платформы и методы, применяемые для разработки, запуска и поддержки веб-приложений. Постоянное развитие интернета и растущий спрос на онлайн-сервисы способствовали появлению множества новых технологий — от языков до баз данных и фреймворков. Внедрение Web-технологий сделало компьютеры доступными для гораздо более широкой аудитории, в том числе для людей, ранее не связанных с информационными технологиями. Число пользователей интернета продолжает увеличиваться за счёт тех, кто использует сеть в повседневной жизни без глубоких технических знаний.

Web-технологии оказали огромное влияние на методы обработки информации и на принципы работы с компьютерной техникой. Долгое время ключевыми характеристиками развития вычислительных систем считались производительность, скорость передачи данных и объём памяти, но проблема удобства взаимодействия с пользователем оставалась на втором плане. Прогресс существенно ускорился лишь тогда, когда интерфейсы стали интуитивно понятными для обычных пользователей, что вызвало массовый рост интереса к компьютерным технологиям.

Под Web-технологиями понимают весь комплекс средств, обеспечивающих функционирование Всемирной паутины (WWW). Поскольку каждый сеанс в сети представляет собой взаимодействие между клиентом и сервером, веб-технологии можно условно разделить на две категории:

1. Технологии клиентской стороны (client-side);
2. Технологии серверной стороны (server-side).

Клиентские технологии включают разработку веб-страниц и интерфейсов (например, HTML, JavaScript, DHTML и др.). Серверные технологии обеспечивают обработку запросов и доступ к базам данных через интернет, используя такие решения, как CGI и PHP.

При запросе информации клиент отправляет серверу специальное сообщение. Сервер обрабатывает его и отправляет в ответ либо запрошенные данные, либо сообщение об ошибке, если ресурс недоступен. Такая модель построения взаимодействия получила название двухзвенной архитектуры или архитектуры "клиент-сервер". Именно на этих принципах основана работа большинства интернет-сервисов, включая Всемирную паутину.

Данная курсовая работа посвящена разработке информационной системы для аэропорта в формате веб-сайта «Аэропорт Онлайн».

Создание онлайн-системы включало в себя следующие этапы:

1. **Определение целей и функций сайта:**  
   На первом этапе был произведён сбор требований к системе. Определены основные функции сайта: отображение расписания рейсов (вылеты и прилёты), предоставление контактной информации, реализация формы авторизации, создание удобной навигации по разделам.
2. **Дизайн:**  
   Следующим шагом стала разработка структуры и внешнего вида сайта. Созданы каркасы страниц: главная страница с приветственным баннером и поиском рейса, страница расписания рейсов с таблицами вылетов и прилётов, страница контактов и страница входа в личный кабинет. Подобраны цветовые схемы (синие и белые тона), шрифты и элементы интерфейса (карточки, таблицы, кнопки).
3. **Разработка:**  
   Реализация проекта выполнена с использованием HTML и CSS.
   * Созданы страницы: index.html, flights.html, contact.html, login.html.
   * Для стилизации использован файл style.css, в котором определены стили для всех элементов: шапки, меню, таблиц, форм, карточек, футера и баннера.
   * Добавлены цветовые бейджи для статусов рейсов (По расписанию, Задерживается, Отменён, Приземлился, В пути), оформленные в виде круглых меток разного цвета.
4. **Тестирование:**  
   После завершения разработки сайт был протестирован в современных браузерах (Google Chrome, Mozilla Firefox, Microsoft Edge) и на различных устройствах (ПК, планшет, смартфон). Проверена корректная работа навигации, отображение таблиц рейсов, правильная смена цветовых тем, а также адаптивность верстки.

WEB-сайт аэропорта — это система, ориентированная на удобство пассажиров и посетителей. Он должен предоставлять сервисы просмотра расписания рейсов, удобный поиск рейсов, актуальную информацию о вылетах и прилётах, а также простую и понятную навигацию.

Таким образом, целью моей курсовой работы является создание сайта для аэропорта, обеспечивающего удобство использования, безопасность передачи данных и интуитивно понятный интерфейс для пользователей.

1. **ТЕОРЕТИЧЕСКАЯ ЧАСТЬ**

**1.1** **Web-программирование**

Web-программирование (или Web-разработка) — это активно развивающееся направление программирования, основная задача которого — создание динамических интернет-приложений. Языки Web-программирования принято делить на две основные категории: клиентские и серверные.

Клиентские языки обрабатываются на стороне пользователя, то есть в его браузере. При этом выполнение скриптов зависит от возможностей браузера: пользователь может отключить их обработку или использовать устаревшую версию браузера, в которой поддержка некоторых языков или их функций отсутствует. В современных браузерах подобные проблемы встречаются редко, а языки программирования обновляются достаточно медленно, поэтому риски несовместимости минимальны. Одной из особенностей клиентского кода является его открытость: любой желающий может просмотреть его, выбрав в браузере пункт меню "Исходный код страницы".

Плюсом клиентских языков является то, что выполнение скриптов не требует обращения к серверу. Например, можно сразу проверить правильность заполнения формы до отправки данных. Однако важно помнить, что средствами клиентских языков нельзя записывать данные непосредственно на сервер. Наиболее распространённым клиентским языком является JavaScript, созданный компанией Netscape совместно с Sun Microsystems. Кроме него, популярен и VBScript, а также современные технологии вроде AJAX, Adobe Flash и Microsoft Silverlight.

Серверные языки программирования предоставляют программистам гораздо больше возможностей. При обращении к веб-странице запрос сначала поступает на сервер, где запускаются необходимые программы, связанные с этой страницей. Лишь после обработки результат отправляется пользователю в виде готового файла. Такие файлы могут иметь расширения HTML, PHP, ASP, Perl, SSI, XML, DHTML, XHTML и другие.

**1.2****HTML и XHTML**

**HTML (HyperText Markup Language)** — это основной язык разметки документов во Всемирной паутине, который принят как стандарт. Большинство веб-страниц в интернете создаются именно с его помощью (или с использованием его расширенной версии — XHTML).

HTML позволяет управлять внешним видом текста и элементов на веб-странице. Можно изменять цвет, стиль, начертание и размер шрифта, создавать заголовки, абзацы, списки и выделять отдельные символы. Одной из ключевых возможностей HTML является создание гипертекстовых ссылок, которые позволяют переходить между страницами и различными частями одного документа.

Также HTML поддерживает формы для ввода данных (например, для отправки информации через сайт), отображение изображений, аудио и видеофайлов — как средствами браузера, так и с помощью внешних приложений вроде Flash или Java-апплетов.

HTML-документы строятся с помощью **тегов** — специальных служебных конструкций. Все теги записываются внутри угловых скобок < >, а для завершения большинства элементов требуется закрывающий тег с прямым слэшем (/), например: <html>...</html>. В HTML допускается использование как строчных, так и заглавных букв в названиях тегов (в отличие от XHTML).

**Пример базовой структуры HTML-документа:**

<HTML>

<HEAD>

<TITLE>Заголовок страницы</TITLE>

</HEAD>

<BODY>

Тело документа

</BODY>

</HTML>

Теги могут быть вложены друг в друга. Некоторые из них (например, <br>) являются самозакрывающимися и не требуют отдельного закрытия.

Атрибуты тегов позволяют задавать их свойства, например цвет текста, размер шрифта, ссылки на изображения. Атрибуты указываются внутри открывающего тега.

**Пример ссылки с атрибутом:**

<a href="http://www.google.com">Перейти на Google</a>

**Основные группы тегов HTML:**

Гиперссылки: <a href="URL" target="\_blank">Текст ссылки</a>, где target определяет, где откроется ссылка (в новом окне, в текущем и т.д.).

**Текстовые элементы:**

Заголовки <h1>…<h6> для выделения текста разного уровня важности.

Абзацы <p>, перенос строки <br>, горизонтальные линии <hr>.

Блоки и области: <div>, <span>, цитаты <blockquote>, текст с сохранением форматирования <pre>.

**Форматирование текста:**

Курсив <i>, жирный <b>, подчёркивание <u>, зачёркивание <s> или <strike>.

Элементы акцента <em>, <strong>.

Изменение размера шрифта <big>, <small>, подстрочный <sub> и надстрочный текст <sup>.

Изменение шрифта <font color="цвет" face="шрифт" size="размер">...</font>.

**Списки:**

Нумерованные <ol> и ненумерованные <ul> списки, элементы списка <li>.

**Изображения:**

Вставка картинок тегом <img src="путь" alt="описание" title="подсказка" width="ширина" height="высота" align="обтекание">.

**Таблицы:**

Создание таблиц через теги <table>, строки <tr>, заголовки столбцов <th>, ячейки <td>.

**Формы ввода данных:**

Создание форм <form>, текстовых полей <input>, больших текстовых областей <textarea>, выпадающих списков <select> и элементов списка <option>.

**Объекты и скрипты:**

Вставка мультимедийных объектов через <embed>, Java-апплетов через <applet>, сценариев через <script>.

**Особые символы HTML:** Некоторые символы нельзя напрямую использовать в коде страницы, например:

&amp; — амперсанд &

&lt; — символ «меньше» <

&gt; — символ «больше» >

**Обязательная структура любой HTML-страницы:**

Она начинается с тегов <html>...</html>, включает секцию заголовка <head>...</head>, и основное содержимое в секции <body>...</body>.

**XHTML (Extensible Hypertext Markup Language)** — это более строгая версия HTML. В XHTML все теги обязательно должны быть закрыты. Так же все теги и атрибуты пишутся только маленькими буквами. XHTML предъявляет более строгие требования к синтаксису (например, нельзя использовать необработанные знаки < и &). При ошибке браузер обязан остановить загрузку страницы. Еще стоит отметить, что XHTML поддерживает расширение языка через пользовательские теги и структуры.

Таким образом, HTML предлагает широкие возможности для форматирования информации на веб-страницах, а XHTML — обеспечивает строгую структуру и уменьшает количество ошибок при создании сайтов.

**1.3 Dynamic HTML**

**Dynamic HTML (DHTML)** — это общее название для технологий, объединяющих язык разметки HTML, каскадные таблицы стилей CSS, скриптовые языки программирования (чаще всего **JavaScript**) и объектную модель документа (**DOM**). Хотя для написания скриптов можно использовать, например, Visual Basic, именно JavaScript стал наиболее распространённым вариантом благодаря своей универсальности и поддержке во всех популярных браузерах.

DHTML позволяет создавать интерактивные веб-страницы, обрабатывать пользовательские запросы и формировать динамическое содержимое без необходимости обращения к серверу или базе данных. С помощью DHTML можно реализовать интерфейсы перетаскивания объектов (Drag'n'Drop), а также создавать игры и другие интерактивные сервисы прямо в браузере пользователя.

Важным преимуществом DHTML является то, что для его работы требуется только браузер, способный обрабатывать HTML, CSS и JavaScript. Никаких дополнительных серверных компонентов не требуется.

**1.3.1 Каскадные таблицы стилей**

**Каскадные таблицы стилей (CSS)** — это технология, предназначенная для управления внешним видом элементов HTML-документа. С помощью CSS задаются такие характеристики, как шрифт, цвет, отступы, размеры и расположение элементов на странице.

Главная идея CSS заключается в разделении структуры документа (HTML) и его оформления (CSS). Благодаря этому подходу веб-разработчики могут легче изменять внешний вид страниц без необходимости вмешательства в их содержимое. Это также помогает уменьшить избыточность кода и облегчает поддержку сайта.

Стили можно подключать несколькими способами:

* **Встроенные стили** — добавляются прямо в HTML-документ внутри тега <style>:

<style type="text/css">

body {

color: green;

}

</style>

* **Внешние таблицы стилей** — подключаются отдельным файлом с помощью тега <link>:

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

* **Inline-стили** — указываются непосредственно в атрибуте style HTML-элемента:

<p style="font-size: 21px; color: green;">Текст</p>

Также каждый браузер имеет собственные встроенные стили по умолчанию.

CSS-правила состоят из **селекторов** и **блока объявлений**:

селектор, селектор {

свойство1: значение;

свойство2: значение;

свойство3: значение;

}

Поскольку одному элементу могут соответствовать несколько стилей, CSS использует принцип каскадности: более специфичные и более поздние правила имеют больший приоритет.

До появления CSS оформление веб-страниц задавалось непосредственно в HTML-коде, что затрудняло поддержку и изменение дизайна. Введение CSS позволило легко применять единый стиль ко многим страницам и быстро адаптировать страницы под разные устройства — например, делать особую версию сайта для печати или для мобильных устройств.

Еще одно важное преимущество внешних CSS-файлов — ускорение загрузки страниц. Браузер загружает таблицу стилей один раз и затем использует её из кэша при переходе между страницами сайта.

Однако CSS имеет и свои недостатки, основной из которых — разное отображение в разных браузерах из-за особенностей их работы с CSS-правилами.

Для работы с CSS доступны специализированные редакторы, например **TopStyle**, который позволяет проверять таблицы стилей на ошибки синтаксиса и совместимость с различными браузерами. Однако чаще всего разработчики используют встроенные редакторы в популярных HTML-редакторах.

**1.3.2 JavaScript**

**JavaScript** — это объектно-ориентированный скриптовый язык программирования, предназначенный для создания динамического поведения на веб-страницах. Несмотря на то, что JavaScript не является официальным стандартом W3C, он занимает важное место в технологии DHTML.

JavaScript имеет синтаксис, похожий на язык C, но отличается рядом особенностей:

* Типы данных определяются динамически.
* Происходит автоматическое приведение типов.
* Память управляется автоматически, что снижает риск утечек.
* Переменные и функции в JavaScript чувствительны к регистру.
* Имена переменных могут содержать буквы, цифры, символ подчёркивания и доллар, но не могут начинаться с цифры.

Из соображений безопасности JavaScript не имеет доступа к файловой системе, сетевым соединениям и не может взаимодействовать с кодом других страниц. Это позволяет безопасно использовать скрипты на любых устройствах — достаточно открыть страницу в браузере.

**Для размещения JavaScript-кода используется тег <script>:**

<script type="text/javascript">

alert('Hello, World!');

</script>

**Также скрипты могут быть вынесены в отдельные файлы и подключаться через атрибут src:**

<script type="text/javascript" src="script.js"></script>

**Благодаря JavaScript разработчики могут:**

* автоматизировать заполнение форм,
* изменять оформление страницы на лету,
* скрывать или показывать элементы,
* управлять поведением веб-приложений на стороне клиента.

Язык довольно прост в освоении и предлагает широкие возможности для создания интерактивных интерфейсов.

Что касается инструментов для разработки на JavaScript, то нет какого-то одного "лучшего" редактора. Многие используют модуль для **Eclipse** или встроенные отладчики в браузерах, которые позволяют удобно проверять и тестировать скрипты прямо во время работы.

* 1. Bootstrap

**Bootstrap** — это свободный и бесплатный фреймворк, основанный на HTML, CSS и JavaScript, который помогает веб-разработчикам быстро создавать адаптивные и красивые сайты и веб-приложения. В состав Bootstrap входят шаблоны для оформления форм, кнопок, навигационных панелей, типографики и других стандартных элементов интерфейса.

Этот фреймворк особенно удобен в следующих случаях:

* когда сайт содержит множество различных страниц,
* когда страницы построены из стандартных компонентов вроде кнопок, таблиц и форм,
* если не планируется полная переработка дизайна в будущем,
* когда нужно быстро и без лишних затрат создать страницы по шаблону.

По сути, **Bootstrap** — это коллекция файлов CSS и JavaScript. После подключения их к проекту разработчик получает доступ к большому количеству готовых классов и компонентов, которые значительно ускоряют процесс создания адаптивной верстки.

Классы Bootstrap можно условно разделить на три основные категории:

* классы для создания сеток и макетов страницы,
* классы для оформления содержимого (текста, изображений, таблиц, кода и т.д.),
* служебные классы для решения вспомогательных задач, таких как скрытие или отображение элементов, изменение отступов, фона и выравнивания текста.

Кроме классов, в Bootstrap включены **компоненты** — это готовые элементы интерфейса: хлебные крошки (отображающие путь навигации по сайту), кнопки, выпадающие списки, подсказки, модальные окна, формы, панели навигации, всплывающие уведомления и многое другое.

Чтобы начать использовать Bootstrap, понадобится текстовый редактор для работы с кодом (например, **Visual Studio Code**, **Atom**, **Sublime Text**) или полноценная среда разработки (**IDE**), а также современный браузер, поддерживающий последнюю версию фреймворка.

Установить Bootstrap можно несколькими способами:

1. **Подключение через CDN (Content Delivery Network)** — достаточно вставить в HTML-шаблон ссылку на необходимые файлы Bootstrap.
2. **Скачивание готовых CSS и JS файлов** и их последующее подключение к проекту локально.
3. **Работа с исходными файлами**, при этом потребуется компилятор CSS (например, для работы с SASS) и автопрефиксатор для кроссбраузерности.
4. **Установка через менеджеры пакетов** — такие как **npm**, **yarn**, **RubyGems**, **NuGet** или **Composer**.

Все эти способы позволяют интегрировать Bootstrap в проект в зависимости от предпочтений разработчика и особенностей рабочего процесса.

**2. ПРАКТИЧЕСКАЯ ЧАСТЬ**

**2.1 Описание предметной области**

**Предметная область** определяется как совокупность теоретических и практических знаний, исследуемых в рамках отдельной научной дисциплины или инженерной задачи. Также под предметной областью понимается определённый сегмент реального мира, изучение которого необходимо для эффективного управления процессами и их последующей автоматизации.  
В рамках данной курсовой работы предметной областью является разработка онлайн-информационной системы для аэропорта, предназначенной для предоставления пассажирам актуальной информации о рейсах, услугах аэропорта и навигации по его территории.

**Архитектура сайта** представляет собой систематизированное распределение информации и построение удобной навигации для обеспечения быстрого доступа пользователей к необходимым данным.  
Грамотно спроектированная архитектура сайта аэропорта позволяет значительно сократить время поиска информации о рейсах, терминалах, услугах и правилах авиаперевозок, что в свою очередь повышает уровень удовлетворённости пассажиров и способствует увеличению популярности аэропорта.  
При проектировании архитектуры необходимо учитывать влияние структуры сайта на восприятие информации пользователями и продвижение предоставляемых услуг в цифровом пространстве. На этапе создания структуры нового сайта или оптимизации существующего особое внимание уделяется акцентированию на важнейших элементах, таких как расписание рейсов, сервисы для пассажиров и навигационные схемы.

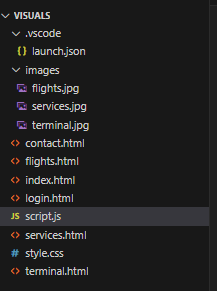
Одним из ключевых компонентов успешного веб-ресурса аэропорта является **пользовательский интерфейс**, обеспечивающий интуитивно понятное взаимодействие между пассажиром и системой. Чёткая организация элементов страницы позволяет быстро находить информацию о рейсах, правилах безопасности, услугах аэропорта и вариантах транспорта до города.

**Для реализации проекта были использованы следующие технологии:**

* **HTML** (HyperText Markup Language) — основной язык разметки веб-страниц, применяемый для создания логической структуры сайта, включая размещение информации о расписании рейсов, описании терминалов и услугах.
* **CSS** (Cascading Style Sheets) — формальный язык описания внешнего вида веб-страниц, разработанных с помощью HTML. С помощью CSS осуществляется стилизация элементов интерфейса: выбор цветовой палитры, размеров шрифтов, отступов, компоновка страниц для различных типов устройств.
* **JavaScript** — язык программирования, обеспечивающий динамическое и интерактивное поведение веб-страниц. Его применение позволяет обновлять данные о рейсах в реальном времени, реализовать формы поиска и бронирования услуг, а также повышать общий уровень удобства использования сайта без необходимости перезагрузки страниц.

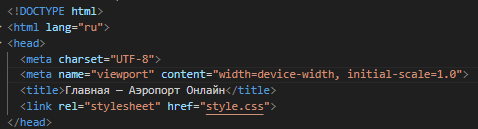
**2.2** **Создание окружения для проекта**

В качестве среды разработки выбран редактор Visual Studio Code. Для проекта создана папка с названием “Visuals”, в которой будут размещаться все необходимые файлы сайта. Поскольку сайт будет состоять из шести страниц, первым создан файлы **index.html**. Далее были созданы две дополнительные папки: **css** и **images**. В папке **css** размещён файл стилей под названием **style.css**, а в папку **images** будут добавляться все изображения, используемые на сайте. После подготовки файловой структуры проект готов к началу разработки.



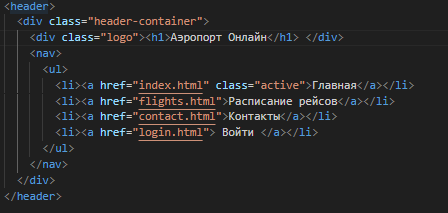
**2.3 Верстка HTML-страницы index.html**

Откроем файл index.html и пропишем стандартную структуру HTML5:



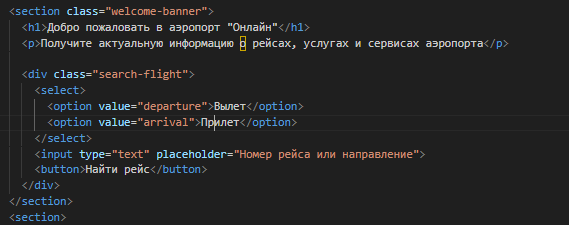
* **<!DOCTYPE html>** сообщает браузеру, что документ написан на HTML5.
* **<html lang="ru">** задаёт язык страницы.
* **<meta charset="UTF-8">** гарантирует корректное отображение кириллицы.
* **<meta name="viewport" content="width=device-width, initial-scale=1.0">** позволяет адаптировать страницу под экраны разных устройств.
* **<link rel="stylesheet" href="style.css">** подключает ваш CSS.

**Навигация (navbar)**

****

* <ul> + <li> + <a> — пункты меню, которые при клике переходят на нужные страницы.

**Баннер с поиском рейса**

****

Сразу после шапки добавляем секцию с ключевым сообщением и формой поиска рейсов:

* Класс .welcome-banner задаёт фоновую картинку или цвет.
* Внутри <div class= “search flight” , можно выбрать между вылетом и прилетом, так же присутсвует поисковая строка для получения информации о рейсе

**Карточки-переходы**

Ниже располагаем блок с карточками для быстрого доступа к ключевым разделам:

A screen shot of a computer program

AI-generated content may be incorrect.

* Каждая карточка — это ссылка <a> с изображением, заголовком <h3> и описанием <p>.
* Класс .card-container выстраивает их в ряд.

**Footer**

В конце файла добавляем футер:

A black background with white text

AI-generated content may be incorrect.

**2.4 Верстка HTML-страницы contact.html**

**Структура и назначение страницы flights.html**

Файл flights.html представляет собой отдельную страницу сайта, предназначенную для отображения расписания вылетов и прилётов в аэропорту. Она оформлена в соответствии с общей стилистикой проекта и включает как статические элементы интерфейса, так и динамически обновляемые данные о рейсах.

**Основная структура документа**

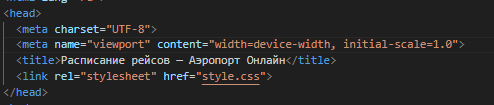
Документ начинается с базовой HTML-разметки:



* **<!DOCTYPE html>** — объявляет тип документа, указывая браузеру, что используется HTML5.
* **<html lang="ru">** — указывает язык содержимого страницы (русский), что важно для корректной обработки текста и доступности.

**Подключение внешних ресурсов**

Внутри тега <head> подключаются основные метаданные и внешние файлы:

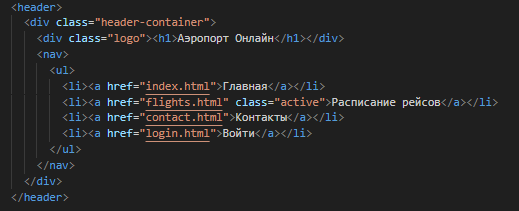


* **<meta charset="UTF-8">** — устанавливает кодировку страницы в UTF-8 для корректного отображения кириллицы и других символов.
* **<title>** — задаёт заголовок страницы, отображаемый на вкладке браузера.
* **<link rel="stylesheet" href="style.css">** — подключает внешний CSS-файл со стилями оформления страницы.

**Структура содержимого страницы (<body>)**

Вся информация для пользователя размещается внутри тега <body>, который включает следующие основные блоки:

**Шапка сайта (<header>)**

****

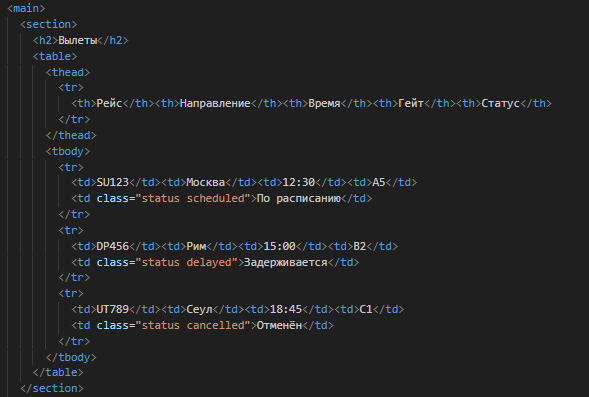
**Пояснение:**

* Блок шапки содержит логотип и навигационное меню.
* Ссылка на текущую страницу (Расписание рейсов) помечена классом active, чтобы выделить её стилистически (например, другим цветом фона в CSS).
* Навигация позволяет быстро перемещаться по основным страницам сайта.

**Основной контент (<main>)**

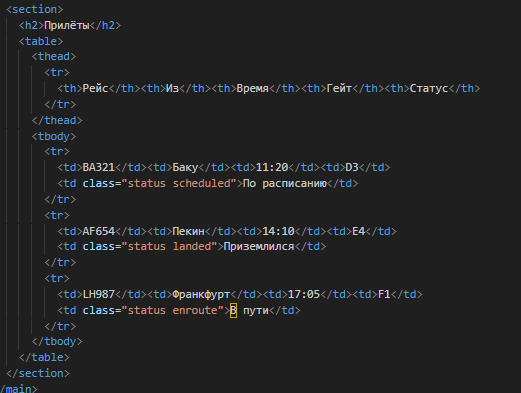
Внутри <main> размещены две ключевые секции:

**Секция "Вылеты"**

****

* Заголовок секции: **"Вылеты"**.
* Таблица содержит информацию о рейсах на вылет: номер рейса, направление, время вылета, номер выхода на посадку (гейт) и текущий статус рейса.
* Статусы рейсов (По расписанию, Задерживается, Отменён) дополнительно стилизованы с использованием CSS-классов (scheduled, delayed, cancelled).

**Секция "Прилёты"**



**Описание:**

* Заголовок секции: **"Прилёты"**.
* Таблица содержит информацию о прибывающих рейсах: номер рейса, пункт вылета, время прибытия, номер выхода и текущий статус рейса.
* Используются аналогичные классы для статусов (scheduled, landed, enroute).

**Оформление статусов рейсов**

Каждому статусу рейса присваивается свой CSS-класс, который определяет цвет фона и внешний вид ячейки.  
Это позволяет пользователю быстро ориентироваться в расписании:

* **По расписанию** — зелёный цвет (.scheduled);
* **Задерживается** — жёлтый цвет (.delayed);
* **Отменён** — красный цвет (.cancelled);
* **Приземлился** — голубой цвет (.landed);
* **В пути** — фиолетовый цвет (.enroute).

Это оформление реализовано в файле style.css, о чём подробно рассказывалось ранее.

**Подвал сайта (<footer>)**

**A black screen with white text

AI-generated content may be incorrect.**

* Подвал страницы содержит юридическую информацию об авторских правах.
* Символ &copy; используется для корректного отображения знака © (копирайт).

**2.5 Верстка HTML-страницы flights.html**

**Структура и назначение страницы contact.html**

Страница contact.html предназначена для предоставления пользователям контактной информации, отображения местоположения аэропорта на карте, а также для организации обратной связи посредством специальной формы.

**Основная структура документа**

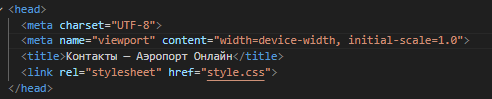
****

Документ начинается стандартной HTML5-разметкой:

* **<!DOCTYPE html>** — указывает браузеру, что используется современный стандарт HTML5.
* **<html lang="ru">** — определяет язык содержимого как русский, что важно для доступности и правильной локализации страницы.

**Подключение внешних ресурсов**

В блоке <head> подключаются основные метаданные и стили:



* **<meta charset="UTF-8">** — задаёт кодировку страницы для поддержки кириллицы и других символов.
* **<meta name="viewport" content="width=device-width, initial-scale=1.0">** — обеспечивает корректное отображение сайта на мобильных устройствах, делая страницу адаптивной.
* **<title>** — задаёт заголовок окна браузера, соответствующий содержанию страницы.
* **<link rel="stylesheet" href="style.css">** — подключает внешний файл стилей, обеспечивая единый дизайн сайта.

**Структура тела документа (<body>)**

**Шапка сайта (<header>)**

**Пояснение:**

A computer screen with text

AI-generated content may be incorrect.

* Шапка включает название проекта (Аэропорт Онлайн) и навигационное меню.
* Текущая страница "Контакты" отмечена классом active, что выделяется соответствующими стилями CSS.

**Основной контент страницы (<main>)**

Содержимое страницы разделено на несколько логических секций (<section>), каждая из которых выполняет свою функцию.

**Контактная информация**

**A screen shot of a computer code

AI-generated content may be incorrect.**

* Содержит основные данные для связи: адрес, телефон и электронную почту.
* Важные слова (Адрес, Телефон, Email) выделены тегом <strong>, что делает их более заметными для пользователя.

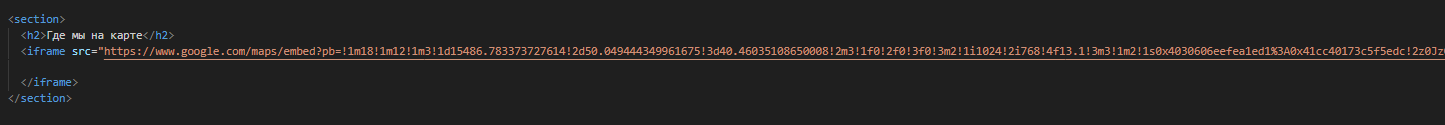
**Часы работы**

**A black screen with white text

AI-generated content may be incorrect.**

* Представлены часы работы организации, что позволяет пользователю заранее знать время приёма и обслуживания.

**Карта местоположения**

****

* Используется встроенный фрейм <iframe>, через который подключается карта Google Maps.
* Свойства карты:
  + width="600" и height="450" — размеры карты на странице.
  + border:0; — убирает рамку вокруг карты.
  + allowfullscreen и loading="lazy" — улучшают производительность и позволяют отображать карту полноэкранно при необходимости.
* Карта отображает точное местоположение аэропорта.

**Социальные сети**

A computer screen with white text

AI-generated content may be incorrect.

Представлены ссылки на страницы аэропорта в социальных сетях. Ссылки разделены вертикальными чертами (|) для удобства восприятия.

**Форма обратной связи**

**A computer screen with text

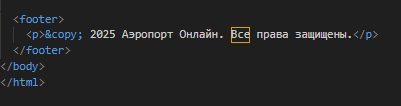
AI-generated content may be incorrect.**

Форма включает следующие элементы:

* **Поле ввода имени** (<input type="text" id="name" required>);
* **Поле ввода электронной почты** (<input type="email" id="email" required>);
* **Текстовое поле для сообщения** (<textarea id="message" rows="5" required>);
* **Кнопка отправки** (<button type="submit">Отправить</button>).

Атрибут required делает все поля обязательными для заполнения. Такая форма позволяет пользователям отправлять свои вопросы или предложения напрямую через сайт.

**Подвал сайта (<footer>)**

****

* Подвал страницы содержит стандартную информацию об авторских правах.

**2.6 Верстка HTML-страницы login.html**

**Структура и назначение страницы login.html**

Страница login.html предназначена для организации процесса авторизации пользователей на сайте "Аэропорт Онлайн". Она представляет собой компактную форму входа с минималистичным дизайном и поддержкой клиентской валидации данных.

**Основная структура документа**

Документ начинается с базовой структуры HTML5:

A black background with white text

AI-generated content may be incorrect.

* **<!DOCTYPE html>** — указывает браузеру, что документ создан в стандарте HTML5.
* **<html lang="ru">** — сообщает браузеру и поисковым системам, что основной язык содержимого страницы — русский.

**Блок <head>**

**A screenshot of a computer code

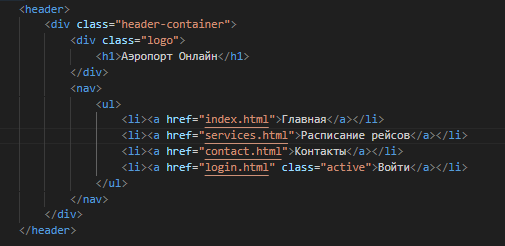
AI-generated content may be incorrect.**

* **<meta charset="UTF-8">** — устанавливает кодировку UTF-8 для корректного отображения всех символов, включая кириллицу.
* **<meta name="viewport" content="width=device-width, initial-scale=1.0">** — обеспечивает корректное масштабирование страницы на мобильных устройствах, способствуя адаптивности дизайна.
* **<title>** — задаёт заголовок страницы "Авторизация", отображаемый во вкладке браузера.
* **<link rel="stylesheet" href="style.css">** — подключает внешний файл таблицы стилей для оформления всех элементов страницы.

**Содержимое страницы (<body>)**

Тело страницы организовано в соответствии с классической структурой: шапка, основной контент и подвал.

**Шапка сайта (<header>)**

****

Шапка включает логотип проекта (Аэропорт Онлайн) и навигационное меню. В меню текущая страница "Войти" выделена классом active, что позволяет визуально подчеркнуть активную ссылку.

Навигация обеспечивает переход на основные страницы сайта.

**Основной контент страницы (<main>)**

**A computer screen shot of text

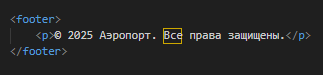
AI-generated content may be incorrect.**

* **Блок .login-form-container**  
  — визуально выделенная область формы входа, стилизованная с помощью CSS (background-color: #fff;, скругление углов, тень).
* **Заголовок формы (<h2>)**  
  — "Вход в систему" — информирует пользователя о назначении формы.
* **Форма авторизации (<form id="login-form">)**  
  Включает три основных элемента:
  + **Поле для ввода логина** (<input type="text">).
  + **Поле для ввода пароля** (<input type="password">).
  + **Кнопку отправки** (<button type="submit">Войти).

Каждый элемент формы сопровождается меткой (<label>), что повышает доступность сайта для пользователей и адаптивных технологий (например, экранных читалок). Поля ввода имеют атрибут required, что позволяет браузеру автоматически проверять, чтобы они не оставались пустыми.

**Подвал сайта (<footer>)**

* Стандартная информация об авторских правах на сайт.



**Подключение скрипта JavaScript**

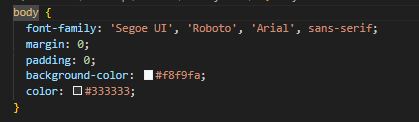
****

Подключение внешнего файла JavaScript (script.js) позволяет добавить интерактивность форме авторизации.Данный скрипт обрабатывает отправку формы, выполняет валидацию данных и перенаправляет пользователя после успешного входа.

**2.7 Использование стилей CSS**

# **Общие стили оформления**

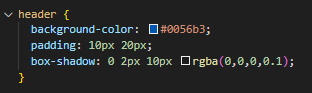
# В начале оформления сайта устанавливаются базовые стили для всего документа. Для этого задаются следующие свойства:



Шрифт устанавливается семейством 'Segoe UI', 'Roboto' или 'Arial', в зависимости от доступности на устройстве пользователя. Также убираются стандартные отступы и поля браузера (margin и padding), задаётся светлый фон и основной цвет текста.

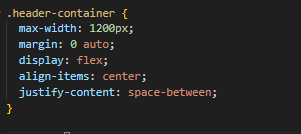
**Оформление шапки сайта**

Для элемента <header> задаются стили, обеспечивающие его визуальное отделение от остальной части страницы:



Таким образом, используется насыщенный синий цвет фона, а также применяется лёгкая тень для создания эффекта возвышенности.

Контейнер внутри шапки оформляется следующим образом:



Это позволяет ограничить максимальную ширину содержимого, выровнять элементы по центру и равномерно распределить их пространство.

Логотип сайта выделяется белым цветом и увеличенным размером шрифта:

A screen shot of a computer code

AI-generated content may be incorrect.

**Стилизация меню навигации**

Навигационное меню реализуется с использованием списка <ul>. Убираются стандартные маркеры и обеспечивается горизонтальное выравнивание элементов:

A screen shot of a computer program

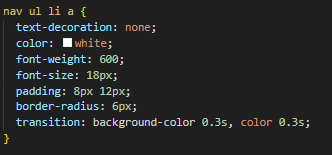
AI-generated content may be incorrect.

Элементы списка получают отступы слева:

A computer code with colorful text

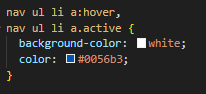
AI-generated content may be incorrect.

Ссылки в меню оформляются следующим образом:

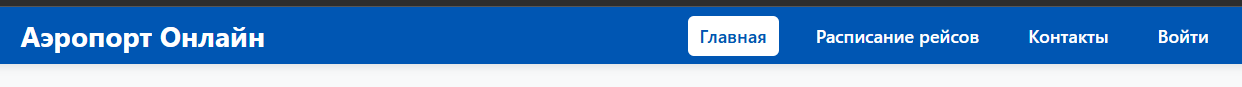


Таким образом обеспечивается отсутствие подчёркивания ссылок и добавляется плавный переход цвета при наведении.

При наведении курсора или активном состоянии ссылки происходит инверсия цвета:

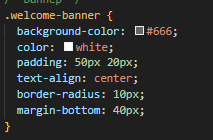


В результате оглавление сайта будет выглядить следущим образом:



**Баннер приветствия**

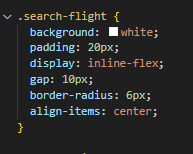
Для оформления приветственного баннера используется следующий CSS-код:



Данный элемент представляет собой блок с фоном тёмно-серого цвета и белым текстом, выравненным по центру. Устанавливаются значительные внутренние отступы для увеличения визуального пространства, а также закругление углов посредством border-radius: 10px. Отступ снизу (margin-bottom) отделяет баннер от следующего содержимого страницы.

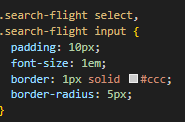
**Форма поиска авиарейсов**

Форма поиска авиарейсов (.search-flight) реализована с использованием следующего стиля:



Контейнер формы представлен как инлайн-флекс контейнер, в котором элементы (поля ввода и кнопка поиска) располагаются в одну строку с промежутками (gap) между ними. Форма имеет светлый фон, скруглённые углы и выравнивание по вертикали за счёт align-items: center.

Поля ввода и выпадающие списки внутри формы имеют следующие стили:



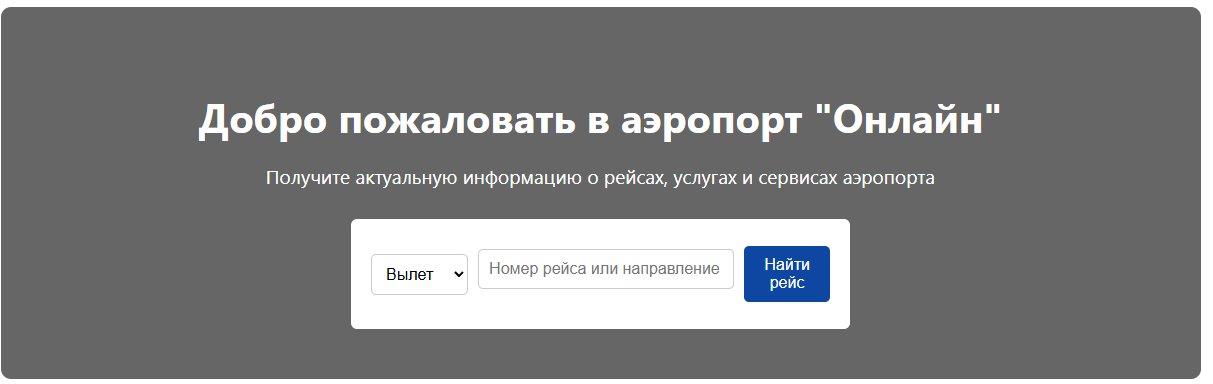
Кнопка поиска выделена синим цветом и имеет эффект затемнения при наведении:

A computer screen shot of a program

AI-generated content may be incorrect.

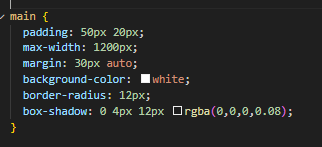
Таким образом, форма поиска органично вписывается в общую концепцию сайта, поддерживая фирменный цветовой стиль.

Конечный вид баннера приветсвия и поиска рейсов:



**Основной контент страницы**

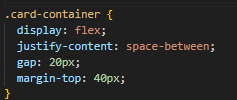
Основной блок сайта стилизован в виде "карточки":



Это обеспечивает аккуратное выделение основного контента за счёт белого фона, скруглённых углов и лёгкой тени.

**Оформление карточек продуктов**

Карточки расположены в ряд с использованием технологии Flexbox:



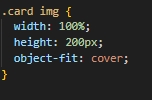
Отдельная карточка имеет следующий стиль:

A computer screen shot of a black screen

AI-generated content may be incorrect.

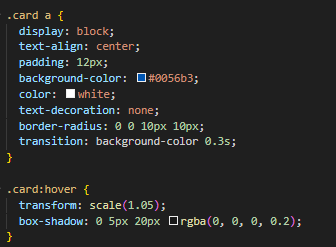
Добавлены скругления, эффект лёгкого подъема при наведении и плавная анимация трансформации.

Изображение внутри карточки адаптируется по размеру:

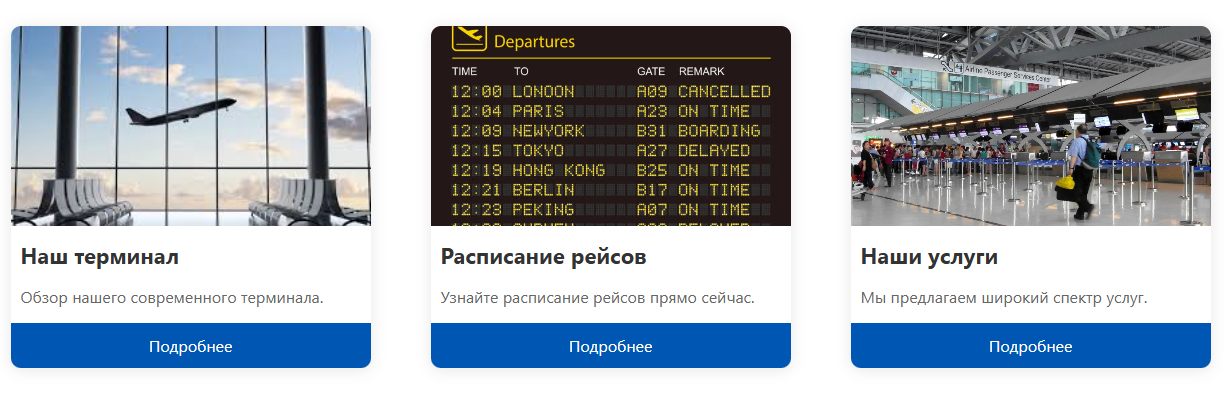


Тексты в карточках имеют компактные отступы и аккуратное цветовое оформление.

Кнопка внутри карточки выделяется цветом и изменяется при наведении:



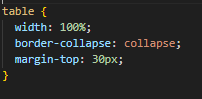
Конечный вид:



**Оформление таблиц данных**

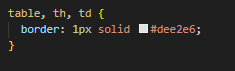
Таблицы, находящиеся в разделе ”Расписание рейсов”, являются важным элементом для структурированного отображения информации на сайте. Для повышения их читаемости и визуальной привлекательности были применены следующие стили CSS.

**Общие стили для таблицы**



* width: 100%; — устанавливает таблицу на всю ширину доступного родительского элемента. Это позволяет таблице равномерно заполнять пространство, обеспечивая адаптивность и предотвращая появление ненужных пустот по краям.
* border-collapse: collapse; — объединяет границы ячеек, устраняя двойные линии между ними. Без этого свойства у таблицы могли бы отображаться двойные границы между ячейками, что ухудшает визуальное восприятие данных.
* margin-top: 30px; — добавляет внешний отступ сверху, отделяя таблицу от предыдущего контента. Это улучшает читаемость страницы, не создавая эффекта слипшегося текста.

**Стилизация границ таблицы**



* Данное правило применяется одновременно к самой таблице (table), заголовкам столбцов (th) и обычным ячейкам (td).
* border: 1px solid #dee2e6; — задаёт тонкую, одинарную границу серого цвета вокруг всех элементов таблицы.  
  Цвет выбран светло-серым (#dee2e6), чтобы визуально отделить ячейки друг от друга, сохраняя при этом лёгкость общего дизайна без тяжёлых визуальных перегородок.

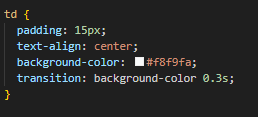
**Оформление заголовков таблицы**

**A screen shot of a computer code

AI-generated content may be incorrect.**

* background-color: #0056b3; — устанавливает насыщенный синий цвет фона для заголовочных ячеек. Это позволяет пользователю легко отличать заголовки от содержимого таблицы.
* color: white; — делает текст заголовков белым, что обеспечивает высокий контраст с фоном и облегчает восприятие информации.
* padding: 15px; — задаёт внутренние отступы, создавая дополнительное пространство вокруг текста и улучшая визуальную структуру заголовков.
* font-size: 16px; — устанавливает оптимальный размер шрифта, соответствующий общему стилю сайта и обеспечивающий удобство чтения.
* text-align: center; — выравнивает текст по центру ячеек заголовков, что делает внешний вид таблицы более аккуратным и симметричным.

**Оформление содержимого ячеек таблицы**



* padding: 15px; — задаёт внутренние отступы в ячейках, что делает текст менее сжатым и увеличивает удобство восприятия данных.
* text-align: center; — выравнивает текст содержимого ячеек по центру, создавая равномерную структуру данных в таблице.
* background-color: #f8f9fa; — устанавливает светло-серый цвет фона для содержимого ячеек, делая таблицу менее перегруженной визуально по сравнению с полностью белым фоном.
* transition: background-color 0.3s; — добавляет плавный переход при изменении цвета фона ячейки, улучшая эстетическое восприятие интерфейса.

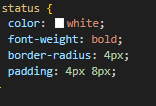
**Эффект при наведении на ячейку**

**A black screen with white text

AI-generated content may be incorrect.**

* При наведении курсора мыши на любую ячейку таблицы цвет её фона меняется на чуть более тёмный оттенок серого (#e2e6ea).
* Этот эффект "подсветки" делает взаимодействие с таблицей более динамичным и наглядным: пользователь интуитивно видит, на какую строку он навёлся.
* Благодаря свойству transition (заданному ранее), смена цвета происходит плавно, что создаёт современный и аккуратный пользовательский опыт.

Особое внимание уделено стилизации таблиц, отображающих статусы рейсов. Для ячеек со статусами (.status) предусмотрено использование цветовых обозначений в зависимости от состояния рейса:

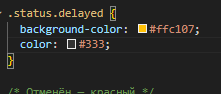


**Статус "по расписанию" (scheduled) окрашен в зелёный цвет:**

**A screen shot of a computer

AI-generated content may be incorrect.**

**Статус "задерживается" (delayed) выделяется жёлтым цветом:**

****

**Статус "отменён" (cancelled) обозначен красным:**

**A screen shot of a computer

AI-generated content may be incorrect.**

**Статус "приземлился" (landed) подсвечивается голубым цветом:**

**A screen shot of a computer

AI-generated content may be incorrect.**

**Статус "в пути" (enroute) представлен фиолетовым цветом**:

A screenshot of a computer

AI-generated content may be incorrect.

Таким образом, пользователь может мгновенно визуально определить состояние рейса благодаря цветовой кодировке.

Результат:

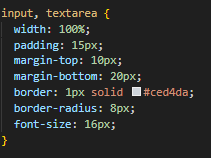
A screenshot of a computer

AI-generated content may be incorrect.

**Стилизация форм ввода данных**

Формы ввода данных на веб-странице играют важную роль в процессе взаимодействия пользователя с сайтом. Для повышения удобства заполнения и эстетической привлекательности формы были оформлены следующим образом.

**Стилизация полей ввода и текстовых областей**



* width: 100%; — задаёт ширину элементов на всю доступную ширину контейнера, что позволяет полям адаптироваться под размеры формы и обеспечивать симметричный внешний вид.
* padding: 15px; — добавляет внутренние отступы внутри полей ввода и текстовых областей, увеличивая площадь активной зоны для ввода текста и улучшая эргономику.
* margin-top: 10px; margin-bottom: 20px; — устанавливает внешние отступы, отделяя поля друг от друга и создавая логичную структуру формы.
* border: 1px solid #ced4da; — задаёт светло-серую границу вокруг полей, обеспечивая чёткое, но ненавязчивое выделение границ полей ввода.
* border-radius: 8px; — делает углы скруглёнными, что визуально смягчает внешний вид формы и соответствует современным тенденциям веб-дизайна.
* font-size: 16px; — обеспечивает оптимальный размер текста для ввода, способствующий удобочитаемости на различных устройствах.

Таким образом, поля ввода и текстовые области становятся интуитивно понятными и удобными для использования.

**Эффект при фокусе на полях ввода**

A screenshot of a computer code

AI-generated content may be incorrect.

**Подробное объяснение:**

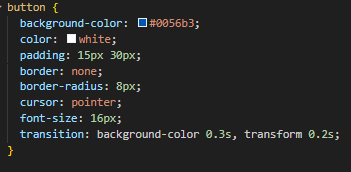
* outline: none; — убирает стандартную браузерную подсветку при фокусе на поле, заменяя её пользовательским стилем.
* border-color: #0056b3; — изменяет цвет границы поля ввода на насыщенный синий, который соответствует общей цветовой гамме сайта.
* box-shadow: 0 0 8px rgba(0,86,179,0.2); — добавляет мягкую синюю тень вокруг поля ввода. Это визуально подчёркивает активное поле, привлекая внимание пользователя, но без излишней агрессивности.

Таким образом, пользователь сразу видит, в каком поле он находится, а форма становится более отзывчивой и приятной для работы.

**Оформление кнопок**

Кнопки являются важнейшими элементами управления на сайте, отвечающими за подтверждение действий пользователей (например, отправку формы).

**Базовое оформление кнопок**



* background-color: #0056b3; — устанавливает фон кнопки насыщенного синего цвета, соответствующего корпоративному стилю сайта.
* color: white; — текст на кнопке окрашивается в белый цвет, обеспечивая высокий контраст и хорошую читаемость.
* padding: 15px 30px; — создаёт внутренние отступы вокруг текста кнопки, увеличивая активную область для нажатия и делая кнопку визуально более крупной.
* border: none; — убирает стандартную границу кнопки браузера, обеспечивая чистый и современный вид.
* border-radius: 8px; — скругляет углы кнопки, гармонируя с оформлением полей ввода.
* cursor: pointer; — изменяет курсор при наведении на кнопку на "указатель" (руку), что интуитивно подсказывает пользователю о возможности взаимодействия.
* font-size: 16px; — задаёт удобный для чтения размер шрифта.
* transition: background-color 0.3s, transform 0.2s; — обеспечивает плавность анимации при наведении, улучшая пользовательский опыт.

**Анимация кнопок при наведении**

A screen shot of a computer

AI-generated content may be incorrect.

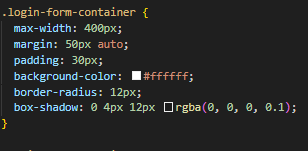
* background-color: #004094; — при наведении цвет фона кнопки становится немного темнее. Это создаёт эффект интерактивности и помогает пользователю понять, что элемент активен.
* transform: translateY(-2px); — при наведении кнопка слегка приподнимается вверх на 2 пикселя. Это имитирует эффект "нажатия" или "отзывчивости", делая интерфейс более живым и современным.

Таким образом, анимация кнопок обеспечивает не только визуальное улучшение интерфейса, но и способствует интуитивному взаимодействию пользователей с сайтом.

**Блок авторизации**

Блок авторизации представлен отдельной секцией с собственным оформлением:

**Базовое оформление контейнера формы**

****

* **max-width: 400px;** — ограничивает максимальную ширину формы 400 пикселями, что обеспечивает её компактность и удобство восприятия как на больших, так и на мобильных экранах.
* **margin: 50px auto;** — задаёт автоматические отступы слева и справа (auto) для центрирования формы по горизонтали, а также верхний и нижний отступ в 50 пикселей для отделения формы от других элементов страницы.
* **padding: 30px;** — устанавливает внутренние отступы, благодаря чему содержимое формы не прилипает к её границам, создавая визуально приятное пространство.
* **background-color: #ffffff;** — задаёт белый цвет фона контейнера, что усиливает акцент на форме по сравнению с общим фоном сайта.
* **border-radius: 12px;** — скругляет углы контейнера, что делает форму более современной и визуально привлекательной.
* **box-shadow: 0 4px 12px rgba(0, 0, 0, 0.1);** — добавляет лёгкую тень вокруг формы, создавая эффект "приподнятости" элемента над остальной частью страницы.

**Стилизация заголовка формы**

**A screen shot of a computer code

AI-generated content may be incorrect.**

* **text-align: center;** — выравнивает заголовок по центру контейнера.
* **color: #0056b3;** — задаёт синий цвет текста заголовка, соответствующий основной цветовой гамме сайта.
* **margin-bottom: 20px;** — создаёт нижний отступ между заголовком и следующими элементами формы, обеспечивая структурированное размещение контента.

**Стилизация меток (label)**

**A screen shot of a computer

AI-generated content may be incorrect.**

* **font-size: 16px;** — устанавливает размер шрифта меток, обеспечивая их удобочитаемость.
* **margin-bottom: 8px;** — задаёт отступ снизу для отделения метки от поля ввода.
* **color: #333;** — применяет тёмно-серый цвет к тексту меток, улучшая контраст и читаемость.
* **display: block;** — делает метку блочным элементом, позволяя корректно размещать её над полем ввода.

**Оформление полей ввода**

**A screen shot of a computer program

AI-generated content may be incorrect.**

* **width: 100%;** — устанавливает ширину поля ввода на всю доступную ширину контейнера, обеспечивая аккуратный внешний вид формы.
* **padding: 15px;** — увеличивает внутренние отступы для создания комфортной зоны ввода текста.
* **margin-bottom: 20px;** — отделяет одно поле ввода от другого.
* **border: 1px solid #ced4da;** — задаёт светло-серую рамку вокруг поля, визуально выделяя область для ввода.
* **border-radius: 8px;** — скругляет углы полей, что соответствует общей стилистике контейнера.
* **font-size: 16px;** — увеличивает читаемость текста, вводимого пользователем.
* **color: #333;** — устанавливает тёмно-серый цвет текста, обеспечивая комфортное восприятие информации.

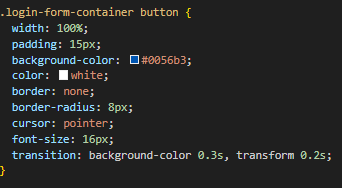
**Анимация полей ввода при фокусе**

**A screen shot of a computer

AI-generated content may be incorrect.**

* **outline: none;** — убирает стандартную браузерную обводку вокруг активного поля ввода.
* **border-color: #0056b3;** — изменяет цвет рамки активного поля на фирменный синий, подчёркивая активное взаимодействие пользователя с формой.
* **box-shadow: 0 0 8px rgba(0,86,179,0.2);** — добавляет лёгкую синюю тень вокруг активного поля, создавая эффект интерактивности и улучшая пользовательский опыт.

**Оформление кнопки авторизации**

****

* **width: 100%;** — растягивает кнопку на всю ширину контейнера.
* **padding: 15px;** — увеличивает внутренние отступы, делая кнопку более заметной.
* **background-color: #0056b3;** — задаёт фирменный синий цвет фона кнопки.
* **color: white;** — устанавливает белый цвет текста для контраста.
* **border: none;** — убирает стандартную рамку вокруг кнопки.
* **border-radius: 8px;** — скругляет углы, соответствуя общей стилистике формы.
* **cursor: pointer;** — при наведении курсор меняется на "указатель" (руку), что визуально подсказывает пользователю о возможности нажатия.
* **transition: background-color 0.3s, transform 0.2s;** — добавляет плавную анимацию для фона и эффекта подъёма при наведении.

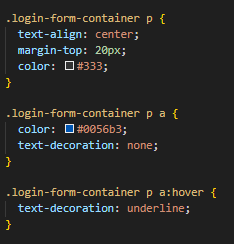
**Эффект при наведении:**

A screenshot of a computer

AI-generated content may be incorrect.

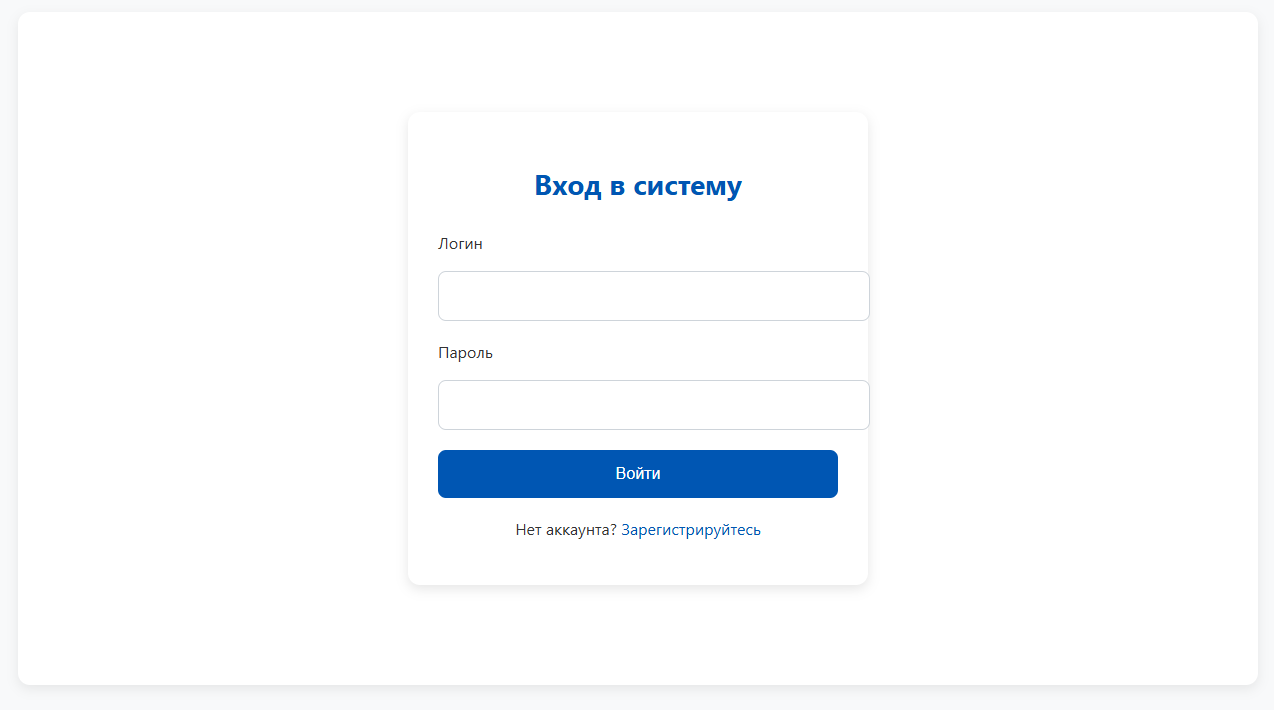
* При наведении фон кнопки становится темнее (#004094), а сама кнопка визуально приподнимается вверх на 2 пикселя (translateY(-2px)), что делает интерфейс более динамичным и отзывчивым.

**Оформление вспомогательных ссылок в форме**

****

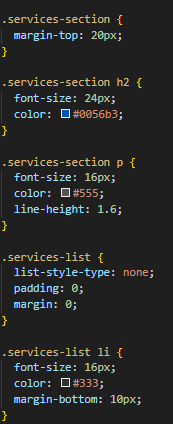
* **Абзац (p)** выравнивается по центру и отделяется от кнопки с помощью верхнего отступа.
* **Ссылки (a)** окрашены в фирменный синий цвет и при наведении подчёркиваются (text-decoration: underline), что традиционно указывает на возможность перехода.

**Результат:**



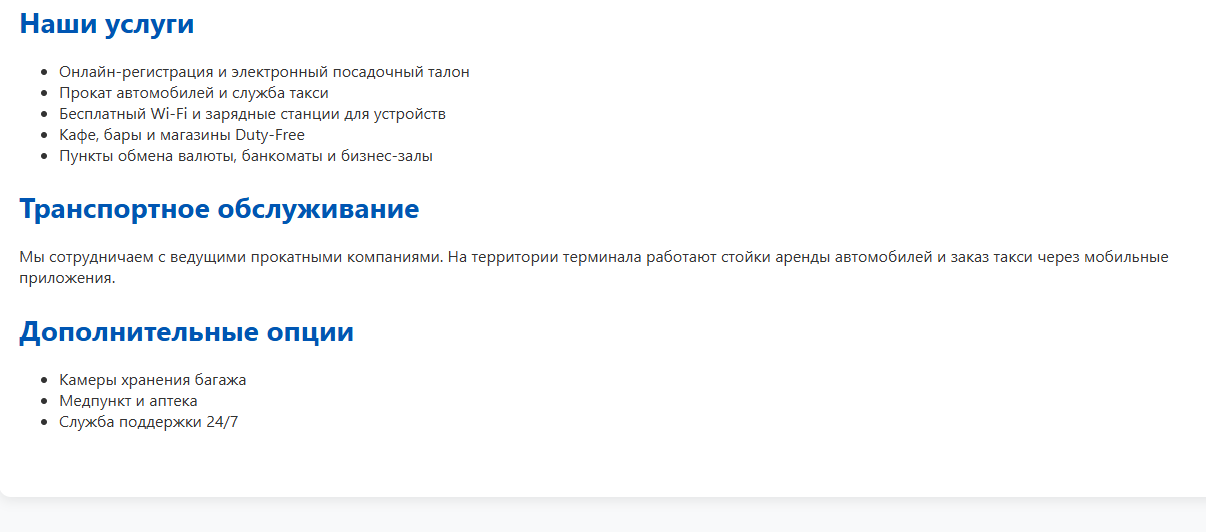
**Секции услуг**

Секции, описывающие предоставляемые услуги, оформлены следующим образом:



Таким образом, текстовый контент в секциях имеет стандартные отступы, оптимальную читаемость и выстроен в логическую структуру. Сами списки услуг оформлены без стандартных маркеров (list-style-type: none).

**Результат:**

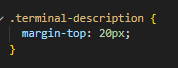


**Оформление секции "Терминал"**

На веб-странице важное место занимает секция, посвящённая описанию терминалов (например, терминалов аэропорта, вокзала или торгового центра). Эта секция предназначена для представления текстовой информации и соответствующих иллюстраций в структурированном и эстетичном виде.

Оформление данной секции в проекте реализовано следующим образом.

**Стилизация текстового блока описания терминала**



* margin-top: 20px; — добавляет внешний отступ сверху для блока .terminal-description, обеспечивая логическое отделение от предыдущего контента и улучшая визуальное восприятие структуры страницы.

**Заголовок описания терминала:**

A screen shot of a computer

AI-generated content may be incorrect.

* font-size: 24px; — увеличивает размер заголовка, делая его визуально доминирующим элементом секции.
* color: #0056b3; — задаёт насыщенный синий цвет текста заголовка, соответствующий общей фирменной палитре сайта.

**Таким образом**, заголовок ясно выделяется и привлекает внимание пользователя.

**Основной текст описания терминала:**

A screen shot of a computer code

AI-generated content may be incorrect.

* font-size: 16px; — задаёт удобочитаемый размер шрифта для основного текста, оптимальный для чтения с различных устройств.
* line-height: 1.6; — увеличивает межстрочный интервал, что значительно улучшает читаемость абзацев, особенно на мобильных устройствах.
* color: #555; — устанавливает средний серый оттенок текста, который обеспечивает хороший баланс между контрастом и мягкостью визуального восприятия.

Таким образом, текст в секции "Терминал" оформлен так, чтобы быть легко читаемым и приятным для глаз.

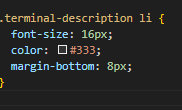
**Стилизация списков описания терминала**

A screen shot of a computer code

AI-generated content may be incorrect.

* list-style-type: square; — задаёт для маркированного списка квадратные маркеры, что визуально отличает списки терминала от стандартных круглых маркеров HTML и делает их более строгими.
* margin-left: 20px; — добавляет отступ слева, обеспечивая чёткое выделение списка относительно основного текста.

**Оформление пунктов списка:**

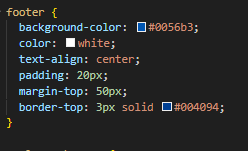


* font-size: 16px; — сохраняет единый размер шрифта для элементов списка, что способствует единству оформления текста.
* color: #333; — делает текст списка чуть темнее основного текста описания, подчеркивая важность каждого пункта.
* margin-bottom: 8px; — добавляет небольшой отступ между элементами списка, улучшая визуальное восприятие и облегчая восприятие информации пользователем.

**Оформление подвала сайта (footer)**

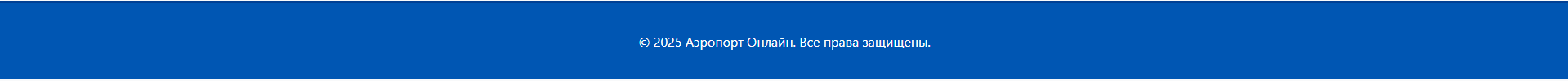
<footer> выполняет важную роль в структуре веб-страницы: он содержит дополнительную информацию, ссылки на условия использования, контакты, авторские права и другие вспомогательные элементы.  
Его оформление должно быть одновременно незаметным (чтобы не отвлекать внимание от основного контента) и в то же время соответствовать общей стилистике сайта.

В разработанном проекте стилизация подвала реализована следующим образом:



* **background-color: #0056b3;**  
  Устанавливает насыщенный синий цвет фона для подвала, который гармонирует с цветовой схемой шапки сайта (header). Это обеспечивает визуальную целостность всего интерфейса.
* **color: white;**  
  Устанавливает белый цвет текста внутри подвала. Белый текст на синем фоне создаёт высокий контраст, способствующий хорошей читаемости информации.
* **text-align: center;**  
  Выравнивает весь текст внутри подвала по центру горизонтально. Центральное выравнивание является стандартной практикой для футеров, где информация обычно представляется в компактной и равномерной форме.
* **padding: 20px;**  
  Добавляет внутренние отступы вокруг содержимого футера. Это предотвращает прилипание текста к краям блока и визуально выделяет текст на фоне.
* **margin-top: 50px;**  
  Создаёт внешний отступ сверху от предыдущего блока (main или любого другого контента страницы), обеспечивая чёткое визуальное разделение основной части сайта и подвала.
* **border-top: 3px solid #004094;**  
  Добавляет верхнюю границу футеру, толщиной в 3 пикселя и более тёмного синего оттенка (#004094), чем фон футера. Эта тонкая полоса отделяет подвал от основного содержимого страницы и визуально усиливает его восприятие как самостоятельного блока.

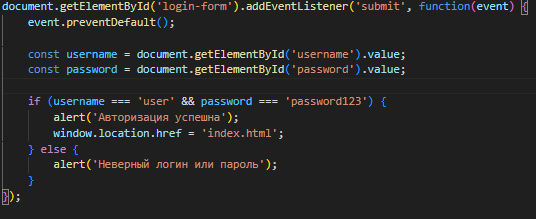
Результат:



**2.8 JavaScript**

**Работа скрипта авторизации пользователей**

Для реализации базовой проверки введённых данных при авторизации на сайте был использован JavaScript-скрипт, представленный в следующем виде:



**Назначение обработчика события отправки формы**

**Пояснение:**

* document.getElementById('login-form') — находит HTML-элемент формы по его атрибуту id="login-form".
* .addEventListener('submit', function(event) {...}) — добавляет обработчик события отправки формы (submit).
* Таким образом, когда пользователь нажимает кнопку "Войти", данная функция будет выполнена.

**Предотвращение стандартного поведения формы**



**Пояснение:**

* По умолчанию при отправке формы браузер перезагружает страницу или перенаправляет её.
* Вызов preventDefault() отменяет это поведение, позволяя обработать данные формы на стороне клиента без перезагрузки страницы.

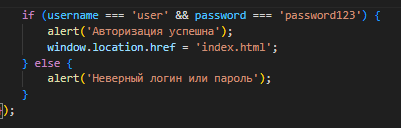
**Получение введённых пользователем данных**

****

**Пояснение:**

* С помощью document.getElementById('username') и document.getElementById('password') получаются элементы формы (поля ввода имени пользователя и пароля).
* .value возвращает значения, введённые пользователем.
* Эти значения сохраняются в переменные username и password для последующей проверки.

**Проверка введённых данных**

****

* Выполняется условная проверка введённых данных:
  + Если имя пользователя равно "user", а пароль — "password123", пользователь считается успешно авторизованным.
  + В противном случае выводится сообщение об ошибке.

**Действия при успешной авторизации**

A screen shot of a computer screen

AI-generated content may be incorrect.

* alert('Авторизация успешна'); — отображает всплывающее сообщение, уведомляющее пользователя об успешной авторизации.
* window.location.href = 'index.html'; — выполняет программное перенаправление на главную страницу сайта (index.html).

**2.12.6 Действия при ошибочной авторизации**

A screen shot of a computer screen

AI-generated content may be incorrect.

* Если введённые пользователем данные не соответствуют ожидаемым значениям, появляется всплывающее окно с сообщением об ошибке.

**3. РЕЗУЛЬТАТ**

В ходе выполнения курсовой работы были изучены и практически применены основные принципы создания веб-сайта:

1. Было разработано несколько веб-страниц с использованием языка разметки HTML, что позволило сформировать структуру и наполнение сайта.
2. Внешнее оформление страниц было реализовано посредством каскадных таблиц стилей (CSS), что обеспечило единое стилистическое оформление и адаптивный дизайн проекта.
3. Создана адаптивная веб-страница с применением медиа-запросов для корректного отображения на различных устройствах.
4. Интерактивные элементы сайта были реализованы с использованием языка программирования JavaScript, что позволило добавить динамическое поведение и повысить удобство взаимодействия пользователей с системой.

В результате проведённой работы был создан современный, функциональный и эстетически привлекательный веб-сайт для аэропорта. Разработанная система соответствует предъявленным требованиям, обеспечивает удобную навигацию, предоставляет актуальную информацию о рейсах и услугах, а также отвечает современным тенденциям веб-дизайна и требованиям к адаптивности.

**4. ЛИТЕРАТУРА**

1. Дакетт Дж. HTML и CSS. Разработка и дизайн веб-сайтов. – М.: Эксмо, 2019.
2. Робсон Э., Фримен Э. Изучаем HTML, XHTML и CSS. – СПб.: Питер, 2018.
3. Фланаган Д. JavaScript. Подробное руководство. – СПб.: Символ-Плюс, 2017.
4. Бьюли А. Изучаем SQL. – СПб.: Символ-Плюс, 2019.

**5. ПРИЛОЖЕНИЯ**

Репозиторий:

<https://github.com/ziyafantaziya/Coursework>