

**Автономное профессиональное образовательное учреждение Вологодской области**  
**«Вологодский колледж связи и информационных технологий»**

**Отчет**

О прохождении практики (учебная, производственная);  
(Вид практики)

Студентом Львов Илья Викторович

Специальность «Информационные системы и программирование» - информационные системы

Курс 4 Форма обучения ОЧНАЯ Группа ИСП – 421 ИС

В Студии интернет-решений «Grampus»  
(Наименование организации)

тема

---

с «17» марта 2025г. по «09» мая 2025г.

Подпись студента \_\_\_\_\_  
(подпись) (расшифровка)

«\_\_» \_\_\_\_\_ 2025г.

Подпись руководителя

Практики от колледжа \_\_\_\_\_ Наталья Вениаминовна Зернова  
(подпись) (расшифровка)

«\_\_» \_\_\_\_\_ 2025г.

Отчет принял

Преподаватель \_\_\_\_\_ Дмитрий Владимирович Репп  
(подпись) (расшифровка)

Оценка (прописью)

«\_\_» \_\_\_\_\_ 2025г.

## СОДЕРЖАНИЕ

ВВЕДЕНИЕ.....	3
РАЗДЕЛ 1. ВЕРСТКА САЙТА .....	5
1.1. Этапы разработки сайта.....	5
1.2. Техническое задание .....	5
1.3 Техника безопасности.....	6
РАЗДЕЛ 2. ХОД ВЫПОЛНЕНИЯ ЗАДАНИЯ.....	8
2.1. Разработка HTML разметки.....	8
2.2. Стилизация сайта .....	11
2.3. Разработка логики сайта .....	13
2.4 Тестирование.....	14
РАЗДЕЛ 3. ПРОЕКТИРОВАНИЕ АВТОМАТИЗИРОВАННОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ ДЛЯ МАГАЗИНА ЭЛЕКТРОНИКИ .....	16
3.1 Введение .....	16
3.2 Моделирование бизнес-процессов.....	16
3.3 Сравнение систем-аналогов .....	18
3.4 Экономическое обоснование разработки .....	19
ЗАКЛЮЧЕНИЕ.....	20
Приложение 1 .....	21

## ВВЕДЕНИЕ

Важно отметить, что прохождение производственной практики в студии интернет-решений «Grampus», расположенной по адресу Вологодская область, г. Вологда, ул. Мальцева, д. 52, 2 этаж, офис 208 (Рисунок 1), был для меня очень ценным опытом.



Рисунок 1. Бизнес-центр

Студия, основанная генеральным директором ИП Николаевым Кириллом Александровичем с 2015 года, специализируется на разработке сайтов и оказывает услуги по продвижению компаний в социальных сетях. Их экспертные знания и навыки помогают клиентам получать больше заказов из интернета. Его компания занимается разработкой сайтов, оказанием услуг по продвижению вашей компании в социальных сетях, которые помогут клиентам получать больше заказов из интернета (Рисунок 2).



Рисунок 2. Вход

Продолжительность прохождения практики в студии составляла четыре недели. За это время мы не только закрепили полученную теоретическую базу, но и существенно расширили свой практический опыт работы. Посредством работы над реальными проектами и выполнением реальных задач, основанных на требованиях клиентов, мы научились более глубоко проникать в суть задачи и проявлять креативность в решении проблем. Это позволило нам развить наши навыки в области верстки веб-страниц, анализа и написания кода, а также эффективного использования инструментов для разработки, таких как «Visual Studio Code» и «Figma».

Отработка основных принципов работы в этих прикладных средах сделала нас более опытными и уверенными в использовании методов разработки и создания веб-сайтов (Рисунок 3).



Рисунок 3. Офис

До места, в котором мы проходили практику достаточно удобно добраться, вход нее находится на втором этаже. Стол, за которым я работал был вполне удобен, а компьютер достаточно производительный (Рисунок 4).

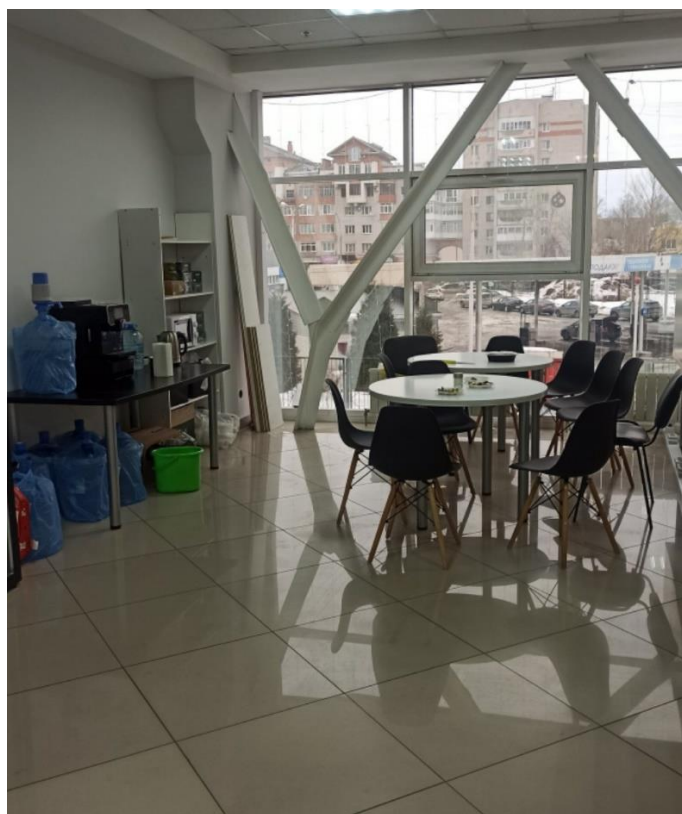


Рисунок 4. Grampus

Прохождение данной производственной практики ведущей студии интернет-решений «Grampus» позволило нам значительно расширить наши знания и практические навыки в области современной верстки веб-сайтов и разработки. Мы глубже поняли востребованность умений веб-верстки, способность кодировать сайты и эффективно использовать инструменты, так как эти навыки являются неотъемлемой частью современного информационного общества и способствуют нашему более успешному и уверенному участию в практических задачах.

## РАЗДЕЛ 1. ВЕРСТКА САЙТА

### 1.1. Этапы разработки сайта

Верстка сайта, которая включает использование языков разметки HTML и CSS, а также языка программирования для написания интерактивной логики JavaScript, является процессом создания веб-страницы с помощью оформления и размещения контента, чтобы он правильно отображался в браузере.

**Задание:** Верстка сайта по дизайну.

Верстка включает в себя следующие этапы:

1. **Создание структуры страницы:** на этом этапе используется язык разметки HTML (Hyper Text Markup Language), который определяет структуру элементов и их иерархию на веб-странице. Заголовки, параграфы, списки, таблицы и другие элементы разметки помещаются в соответствующие теги для определения их семантики и отношений друг с другом.

2. **Оформление и стилизация страницы:** здесь применяется CSS разметка для задания внешнего вида элементов страницы. CSS определяет цвета, шрифты, размеры, отступы, рамки и другие атрибуты стилизации. Стили могут быть заданы для отдельных элементов, классов или идентификаторов, а также использоваться селекторы и правила для группировки стилей.

3. **Адаптивная верстка:** в данном случае, адаптивность означает, что верстка сайта настраивается для оптимального отображения на различных устройствах и экранах. Используется подход «RWD» (Responsive Web Design), где задаются медиа-запросы, которые позволяют странице перестраиваться и адаптироваться к разным размерам экранов.

4. **Работа с медиа-элементами:** на этом этапе добавляются изображения, видео и аудио на веб-страницу. Для оптимизации загрузки страницы, изображения могут быть оптимизированы по размеру и сжаты. Теги, такие как `<img>` для изображений или `<video>` для видео, используются для встраивания медиа-элементов на страницу.

5. **Добавление интерактивности:** здесь применяется язык программирования JavaScript для создания интерактивных функций и эффектов на веб-странице. Этот язык позволяет реализовывать анимации, валидацию форм, выпадающие меню, слайдеры и многое другое.

6. **Оптимизация загрузки страницы:** верстка также включает в себя методы оптимизации загрузки страницы для улучшения производительности и скорости ее загрузки.

7. **Тестирование и отладка:** на данном этапе осуществляется тестирование верстки на различных браузерах и устройствах

### 1.2. Техническое задание

Задание 1:

1. Сверстать сайт, используя HTML, CSS по макету (Рисунок 5);

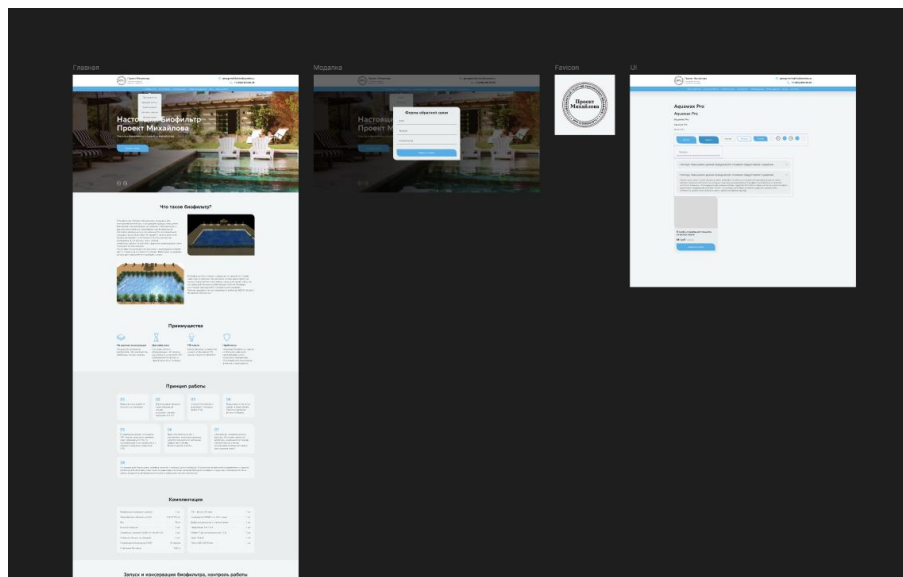


Рисунок 5. Макет первого сайта

Дополнительное задание:

1. Адаптация для всех разрешений устройств (Ноутбуки, планшеты, смартфоны) (1920px - 320px);
2. Реализовать модальные окна, используя библиотеку Fancybox (Кнопки «Оставить заявку»);

Задание 1:

1. Сверстать сайт, используя HTML, CSS по макету (Рисунок 6);

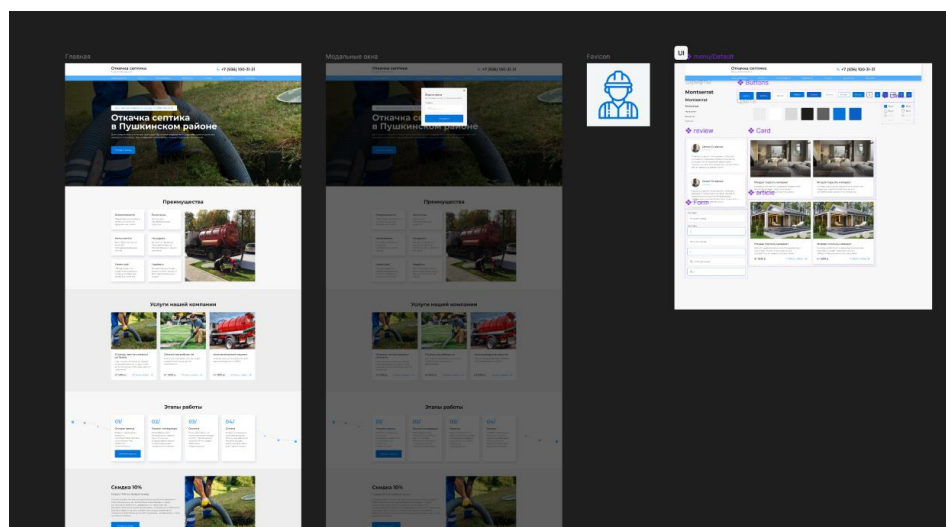


Рисунок 6. Макет второго сайта

Дополнительное задание:

1. Адаптация для всех разрешений устройств (Ноутбуки, планшеты, смартфоны) (1920px - 320px);
- Реализовать модальные окна, используя библиотеку Fancybox (Кнопки «Оставить заявку»);

### 1.3 Техника безопасности

#### 1. Перед началом работы:

- 1.1 Проверить исправность электропроводки, розеток и кабелей компьютера.

1.2 Убедиться, что рабочее место хорошо освещено, а монитор не создает бликов.

1.3 Отрегулировать кресло и монитор для комфортной работы (спина прямая, экран на уровне глаз).

**2. Во время работы запрещается:**

2.1 Работать на неисправном оборудовании (перегревается, искрит, шумит).

2.2 Чистить компьютер или периферию при включенном питании.

2.3 Самостоятельно ремонтировать технику без соответствующих навыков.

2.4 Ставить рядом с ПК жидкости (чашки, бутылки с водой).

2.5 Работать с мокрыми руками или касаться металлических конструкций (батарей, труб).

2.6 Курить или принимать пищу за рабочим столом (риск попадания крошек и влаги в клавиатуру).

**3. В аварийных ситуациях:**

3.1 Немедленно отключить ПК от сети.

3.2 Если обнаружен оголенный провод – предупредить окружающих и не прикасаться к нему.

3.3 Отключить питание, использовать огнетушитель (СО<sub>2</sub> или порошковый).

3.4 Отключить электропитание, оказать первую помощь, вызвать скорую.

**4. После работы:**

4.1 Закрыть все программы и корректно выключить компьютер.

4.2 Отключить питание (если не требуется оставлять ПК в режиме ожидания).

4.3 Привести рабочее место в порядок.

## РАЗДЕЛ 2. ХОД ВЫПОЛНЕНИЯ ЗАДАНИЯ

### 2.1. Разработка HTML разметки

#### Выполнение первого сайта:

HTML является основным строительным блоком веб-страниц и предоставляет стандартизированный синтаксис разметки. Он обеспечивает доступность, переносимость и совместимость веб-сайтов между разными браузерами и устройствами. HTML играет ключевую роль в создании информационной структуры веб-страниц и формировании пользовательского опыта при просмотре веб-сайтов.

**<head>**: элемент метаданных документа.

HTML-элемент **<head>** содержит дополнительную машиночитаемую информацию (metadata) о документе, например его заголовок, скрипты и страницы стилей. Примечание: **<head>** в основном содержит информацию для машинной обработки, а не для восприятия человеком.

В данном элементе я указал:

1. Значение **<Viewport>** указывающие что ширина страницы равна ширине экрана и масштабирование в 1 единицу, то есть без масштабирования (Рисунок 7).
2. Заголовок для вкладки в браузере.
3. Путь к основному файлу с стилями CSS;

```
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Проект Михайлова</title>

  <link rel="icon" href="favicon.png">
  <link rel="stylesheet" href="index.css">
  <link rel="stylesheet" href="adaptive.css">
  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/@fancyapps/ui@5.0/dist/fancybox/fancybox.css" />

  <script src="https://cdn.jsdelivr.net/npm/@fancyapps/ui@5.0/dist/fancybox/fancybox.umd.js"></script>
  <script type="module">Fancybox.bind("[data-src]", { closeButton: false })</script>
</head>
```

Рисунок 7. Элемент метаданных документа **<head>**

**<body>**: элемент, которые представляет собой контент

В стандарте HTML может быть только один элемент **<body>**, который определяет основное отображаемое содержимое веб-страницы.

Элемент **<body>** содержит все отображаемые на странице элементы, такие как контейнеры **<div>**, абзацы **<p>**, заголовки **<h1>** и **<h2>**, растровые **<img>** и векторные **<svg>** изображения, ссылки **<a>** и другие элементы, показывая их на экране пользователя в порядке расположения их в разметке или другими различными методами указанными в файле с стилями.

Современные браузеры поддерживают потоковый контент такой как изображения **<img>**, видео **<video>** и другой контент, который нужно загружать. С помощью потокового контента можно создавать веб-страницы с аудио- и/или видеопроигрывателями, которые могут воспроизводить контент до завершения загрузки файла. Это полезно для оптимизации



производительности и улучшения пользовательского опыта, поскольку позволяет пользователям начать просмотр контента до того, как весь файл будет загружен.

Открывающий тег элемента `<body>` может быть пропущен в случае, если первое, что находится внутри этого элемента, не является пробелом, комментарием, элементом `<script>` или элементом `<style>`. В таком случае, браузер автоматически вставляет открывающий тег `<body>` перед этим контентом.

Это правило позволяет упростить написание HTML кода и сделать его более лаконичным. Однако рекомендуется всегда явно указывать открывающий и закрывающий теги для элемента `<body>` для более ясного и читаемого кода.

В специальном элементе `<header>` указываются (Рисунок 8):

1. Кнопки навигации;
2. Почта для обратной связи;
3. Номер телефона для связи.

```
<header>
  
  <section>
    <span>
      <h3>Проект Михайлова</h3>
      <h6>Самозанятый Георгий<br>Рафаилович Михайлов</h6>
    </span>
    <span id="header-full-contacts">
      <h3>georgymikhailov@yandex.ru</h3>
      <h3>+7 (999) 999-99-99</h3>
    </span>
    <span id="header-mobile-contacts">
      <h3>georgymikhailov@yandex.ru</h3>
      <h3>+7 (999) 999-99-99</h3>
    </span>
  </section>
</header>
```

Рисунок 8. Элемент `<header>`

### Выполнение второго сайта:

HTML — это не просто язык разметки, а инструмент для создания доступных и индексируемых веб-ресурсов. В проекте особое внимание уделялось:

**Семантическим тегам:** `<header>`, `<nav>`, `<main>` вместо универсальных `<div>` для улучшения SEO и поддержки скринридеров.

**Микроразметке:** Добавление атрибутов `aria-label` для кнопок навигации, что повышает доступность для пользователей с ограниченными возможностями.

**Оптимизации загрузки:** Использование атрибута `loading="lazy"` для изображений вне зоны видимости, что снижает начальную нагрузку страницы. (Рисунок 9)

```

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Откачка септика</title>

  <link rel="icon" href="favicon.png">
  <link rel="stylesheet" href="index.css">
  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/@fancyapps/ui@5.0/dist/fancybox/fancybox.css" />

  <script src="https://cdn.jsdelivr.net/npm/@fancyapps/ui@5.0/dist/fancybox/fancybox.umd.js"></script>
  <script
    type="module">Fancybox.bind("[data-src]")</script>
</head>

```

Рисунок 9. Элемент метаданных документа **<head>**

Элемент **<body>** представляет собой фундаментальную часть HTML-документа, содержащую весь визуальный контент веб-страницы. Этот обязательный элемент определяет основную область отображения, где размещаются все компоненты интерфейса, доступные пользователю. Современные веб-браузеры обрабатывают содержимое body с использованием передовых технологий рендеринга, обеспечивая плавное отображение контента даже при постепенной загрузке ресурсов.

Особое внимание при работе с body уделяется организации структуры документа. Профессиональные разработчики предпочитают явное указание тегов, несмотря на возможность их опускания в некоторых случаях. Такой подход повышает читаемость кода и снижает вероятность возникновения ошибок интерпретации. Внутри body располагаются все видимые элементы страницы, чье отображение может контролироваться как средствами HTML, так и через CSS-стили.

Элемент **<header>** выполняет роль важной навигационной и информационной секции в структуре веб-страницы. В рассматриваемом проекте он спроектирован как многофункциональный блок, объединяющий ключевые элементы управления и важную контактную информацию. Его реализация учитывает современные требования к адаптивности и доступности, обеспечивая корректное отображение на различных устройствах и платформах.

Техническая реализация этих элементов включает комплексный подход, сочетающий семантическую разметку, оптимизированные стили и продуманную логику взаимодействия. Особое внимание уделено производительности и скорости загрузки, что достигается за счет применения современных методов веб-оптимизации. Архитектура этих компонентов разработана с учетом возможностей масштабирования и дальнейшего развития проекта.

В специальном элементе **<header>** указываются (Рисунок 10):

1. Заголовок;
2. Иконка;
3. Номер телефона для связи.

```

<header>
  <div>
    <span>
      <h2>Откачка септика</h2>
      <h5>В Пушкинском районе</h5>
    </span>

    <h2>+7 (936) 100-31-31</h2>
  </div>
</header>

```

Рисунок 10. Элемент `<header>`

## 2.2. Стилизация сайта

CSS (Cascading Style Sheets) — это язык стилей, который используется для описания и определения внешнего вида документов, написанных с использованием языка разметки, например, HTML. С его помощью можно управлять стилями, цветами, шрифтами, размерами, расположением элементов на странице и другими атрибутами, чтобы создавать эстетически приятные и функциональные веб-страницы.

CSS декларирует стили, которые применяются к конкретным элементам разметки веб-страницы. Декларации стилей состоят из селекторов и объявлений. Селекторы указывают, к каким элементам применяются стили, а объявления содержат инструкции отображения, например, цвет фона, размер шрифта или отступы. Набор правил CSS позволяет гибко задавать стиль и расположение элементов на странице.

CSS также может применяться к другим типам документов XML, таким как SVG (Scalable Vector Graphics) – формат для создания векторной графики, и XUL (XML User Interface Language) – для разработки пользовательского интерфейса. CSS обеспечивает согласованность и отделение оформления от контента в этих документах, позволяя управлять их визуальным представлением.

CSS является важной составной частью веб-разработки и способствует созданию эстетически приятных, функциональных и современных веб-страниц.

### Выполнение первого сайта:

Атрибут «*display: flex*» вместе с атрибутом «*justify-content: end*» указывает браузеру, что элемент должен отображаться как специальный «flex-контейнер», который был добавлен не так давно, относительно истории разметки CSS, цель которого упростить адаптацию сайта путем «flex» (гибкого) позиционирования элементов с помощью специальных атрибутов с удобными для адаптации значениями. В атрибуте указан «*justify-content*» указывающий, что элементы по горизонтали в контейнере должны располагаться по горизонтали в конце самого контейнера.

Значение атрибута «*padding-right: 10px*» указывает элементы что его содержимое должно иметь отступ справа шириной в 10 пикселей, сам элемент не изменяет своих размеров и положения.

В самом стиле (Рисунок 11) указан атрибут **«font-weight»** который устанавливает шрифты для всех элементов в значении **«bold»**, который был указан в макете сайта и был импортирован из внешнего файла и подключен в начале файла со стилями.

```
font-family: 'Aquawax Pro';
src: url('fonts/AquawaxPro-Bold.eot');
src: local('Aquawax Pro Bold'), local('AquawaxPro-Bold'),
    url('fonts/AquawaxPro-Bold.eot?#iefix') format('embedded-opentype'),
    url('fonts/AquawaxPro-Bold.woff2') format('woff2'),
    url('fonts/AquawaxPro-Bold.woff') format('woff'),
    url('fonts/AquawaxPro-Bold.ttf') format('truetype');
font-weight: bold;
font-style: normal;
```

Рисунок 11. Атрибут **«font-weight»**

Атрибуты **«padding»** указывают что все элементы по умолчанию будут иметь внутренний и внешний отступ в 0 единиц (пикселей). Он может применяться для всех сторон элемента (верх, право, низ, лево) или применяться к каждой стороне индивидуально, в данном случае указаны все стороны одной цифрой.

#### **Выполнение второго сайта:**

В текущем проекте стилевое оформление реализовано в отдельном файле объемом 800 строк кода. Ключевые аспекты стилизации включают:

Использование Flexbox-модели с параметром justify-content: end, который обеспечивает выравнивание элементов по правому краю контейнера. Эта современная технология макетирования значительно упрощает создание адаптивных интерфейсов.

Точная настройка пространственного расположения элементов через свойство padding-right: 10px, создающее фиксированный отступ содержимого от правой границы элемента без изменения его основных размеров.

Единая типографическая система, основанная на шрифте Ubuntu, который был специально импортирован для соответствия дизайн-макету. Глобальное применение шрифта через свойство font-family обеспечивает визуальную согласованность всех текстовых элементов интерфейса.

CSS продолжает развиваться, предлагая все новые возможности для создания сложных, визуально привлекательных и технически совершенных веб-интерфейсов (Рисунок 12). В данном проекте грамотное применение CSS-технологий позволило достичь оптимального баланса между эстетикой и функциональностью конечного продукта.

```
* {
padding: 0;
margin: 0;
color: var(--MainText);

box-sizing: border-box;
}
```

Рисунок 12. Селектор «\*»

Атрибуты «*padding*» указывают что все элементы по умолчанию будут иметь внутренний и внешний отступ в 0 единиц (пикселей). Он может применяться для всех сторон элемента (верх, право, низ, лево) или применяться к каждой стороне индивидуально, в данном случае указаны все стороны одной цифрой.

### 2.3. Разработка логики сайта

Каждый сайт представляет из себя не только текст с картинками, но и интерактивную логику такую как: кнопки, видео, анимации и т.д., которые могут реагировать на действия пользователя, изменять свое состояние в зависимости от разных условий, отображать динамически изменяемую информацию.

Обычно логика сайта создается при помощи языков программирования, раньше так и было, но в современности даже на языке разметке CSS можно создавать простые эффекты такие как: изменение отображения элемента при наведении, зажатии, отпуске мыши, отображение простых видео (слайд-шоу или GIF анимации), простые анимации и другие эффекты, не требующие сложной логики и много зависимостей.

#### Выполнение перового сайта:

Язык разметки CSS позволяет реализовать логику страницы при помощи «псевдо-элементов», например изменение стиля элемента осуществляет при помощи псевдо-элемента «*hover*» который обязательно должен идти после селектора элемента, на который данная логика будет применима (Рисунок 13).

```
button:hover {
background-color: var(--Hover);
}
```

Рисунок 13. Изменение стиля при наведении мыши

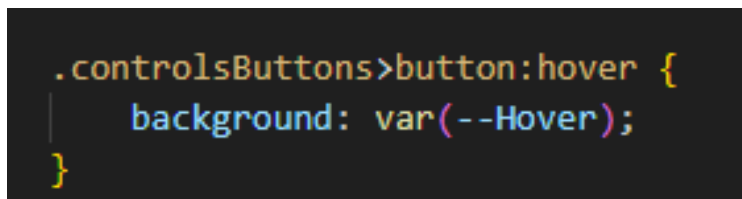
Для создания более сложной логики с применением условий, изменение большого количества элементов или отображения элементов, полученных извне браузера, используется в большинстве случаев скриптовый язык «JavaScript», он не нуждается в компиляции, сложном изучении и код на нем очень компактен. Он применяется в большинстве сайтов, но также есть аналоги от разных компаний такие как «Cmm», «РНР» и т.д., которые не получили такой

популярности, хотя PHP все еще довольно часто применяется, в основном для поддержки старых сайтов.

Сайт был успешно сверстан при помощи разметки HTML, стилизован при помощи CSS и наделен интерактивными элементами по большей части с помощью псевдо-элементов CSS.

### **Выполнение второго сайта:**

CSS предоставляет мощные инструменты для создания интерактивных элементов через псевдо-классы. Например: `hover` позволяет изменять стили при наведении курсора (Рисунок 14). Этот подход эффективен для простых визуальных эффектов, не требующих сложной логики.



```
.controlsButtons>button:hover {  
    background: var(--Hover);  
}
```

Рисунок 14. «hover» эффект

Для реализации более сложного взаимодействия, такого как динамическая загрузка данных или обработка форм, применяется JavaScript. Этот язык сценариев отличается простотой интеграции и широкими возможностями, что делает его стандартом для современной веб-разработки. В отличие от альтернативных технологий вроде PHP, которые преимущественно используются для серверной части, JavaScript выполняется непосредственно в браузере, обеспечивая мгновенную реакцию интерфейса.

В данном проекте интерактивность достигнута за счет комбинации:

CSS-псевдоэлементов для базовых эффектов

Чистого JavaScript для сложных сценариев

Семантической HTML-разметки как основы структуры

Такой подход позволил создать отзывчивый интерфейс с оптимальным соотношением производительности и функциональности.

## **2.4 Тестирование**

Цели:

1. Корректное отображение;
2. Совместимость с различными устройствами;
3. Соответствие стандартам;
4. Оптимизация производительности.

Выполнил отладку кода с помощью инструмента «Отладка» в VisualStudio:

1. **Подготовка проекта:** Сначала я создаю новый проект в Visual Studio и добавляю в него HTML-файл, где буду писать код.

2. **Написание кода:** После этого я открываю HTML-файл и пишу необходимый HTML-код, необходимый для реализации функционала страницы.

3. **Запуск проекта:** Для тестирования созданного кода я запускаю проект, нажав клавишу F5 или выбрав команду «Запустить отладку» из меню.

4. **Использование инструментов разработчика:** После запуска приложения я открываю инструменты разработчика в браузере (например, нажимаю F12), чтобы проверить структуру HTML-кода и просмотреть возможные ошибки (Рисунок 15).

5. **Отладка JavaScript-кода:** Если в коде используется JavaScript, я обращаюсь к вкладке «Консоль» для просмотра ошибок и отладочной информации. Устанавливаю точки останова в JavaScript-коде, что позволяет пошагово выполнять код и анализировать значения переменных.

6. **Корректировка ошибок:** На основе полученной информации я вношу необходимые изменения в код, затем обновляю страницу и продолжаю отладку до тех пор, пока все ошибки не будут устранены.

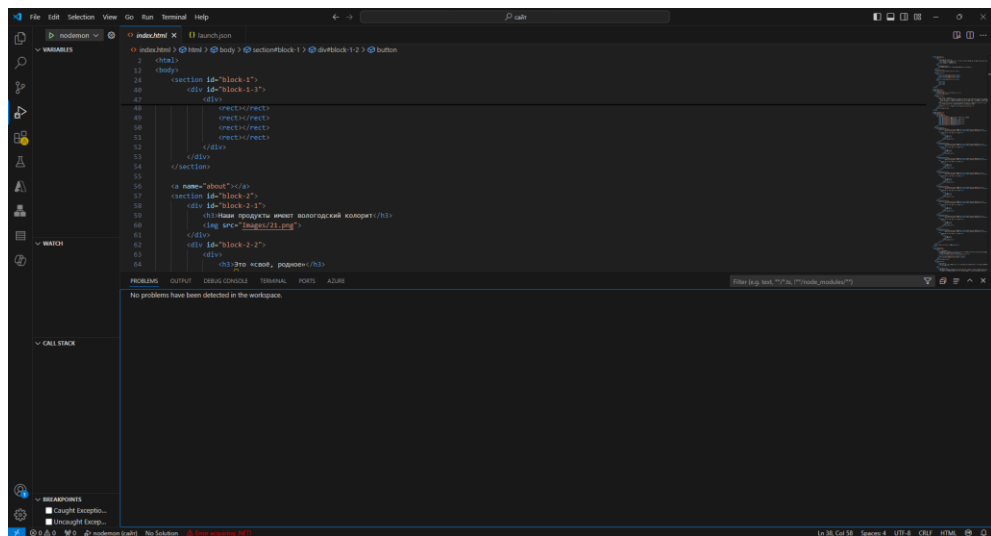


Рисунок 15. Использование консоли

## РАЗДЕЛ 3. ПРОЕКТИРОВАНИЕ АВТОМАТИЗИРОВАННОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ ДЛЯ МАГАЗИНА ЭЛЕКТРОНИКИ

### 3.1 Введение

Современный рынок электроники неуклонно растет, и онлайн-продажи занимают все более значимое место в этой сфере. В данном дипломном проекте рассматривается разработка веб-сайта для онлайн-магазина, специализирующегося на продаже аксессуаров Apple, таких как наушники, зарядные устройства и джойстики. Целью проекта является создание удобной и функциональной платформы для пользователей, где они смогут легко находить и приобретать необходимую продукцию. В процессе работы над проектом будут проанализированы требования к системе, выполнено проектирование, а также сравнение с аналогичными решениями на рынке.

Основными требованиями к разрабатываемому сайту являются:

1. Верстка сайта, используя HTML, CSS по макету;
2. Адаптация для всех устройств;
3. Реализовать модальные окна, используя библиотеку Fancybox;

### 3.2 Моделирование бизнес-процессов

Главная цель «Use-case» диаграммы — предоставить ясное описание того, как система будет взаимодействовать с пользователями (Рисунок 16), что позволяет заказчику, конечным пользователям и разработчикам совместно анализировать и обсуждать проектируемую или уже существующую систему.



Рисунок 16. Use-case диаграмма



Главная цель диаграммы последовательности — иллюстрировать взаимодействие между объектами системы (Рисунок 17) на протяжении времени. Она предоставляет визуальное представление того, как объекты взаимодействуют друг с другом с помощью сообщений, а также показывает порядок, в котором эти взаимодействия происходят.

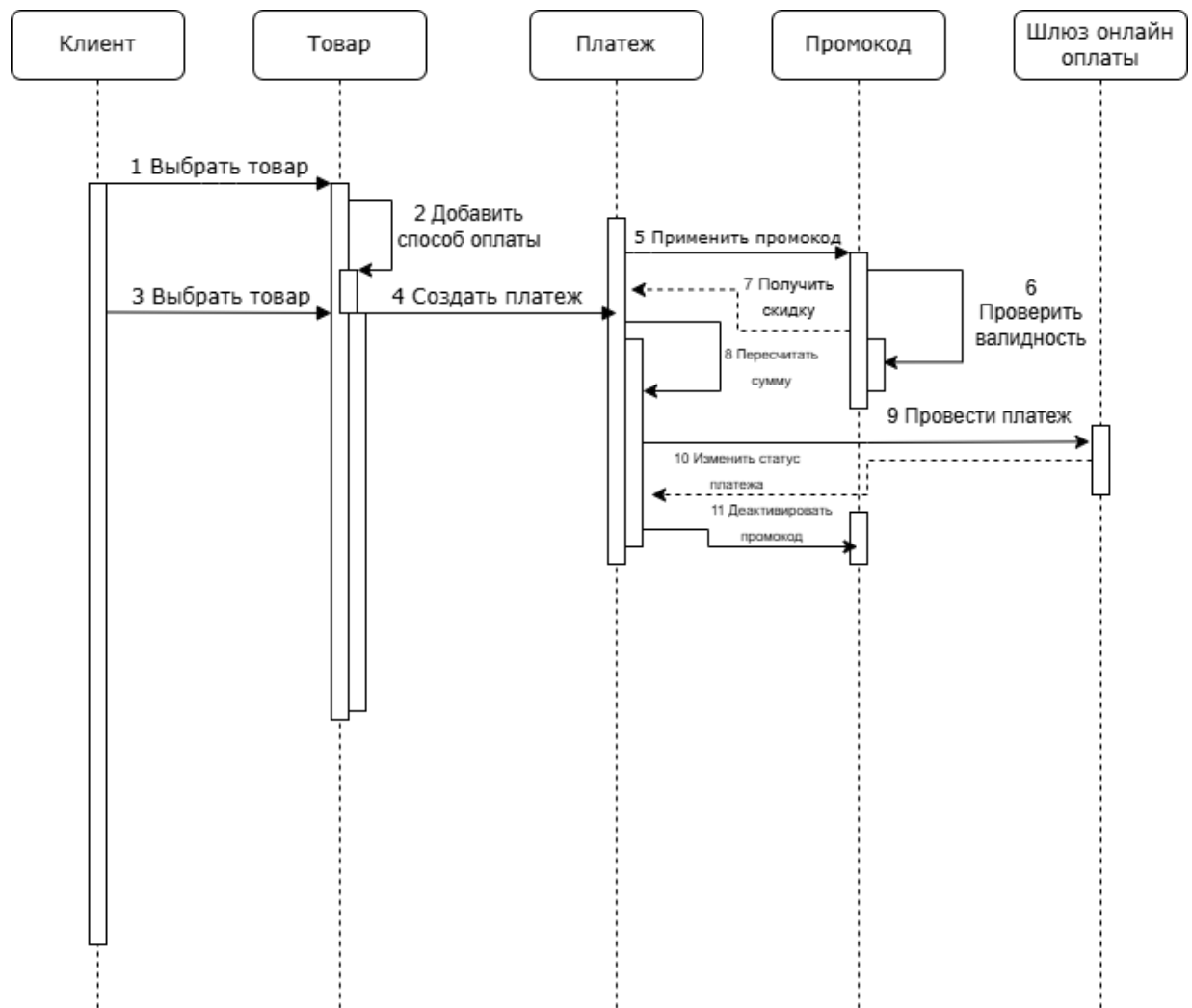


Рисунок 17. Диаграмма последовательности

Основная цель UML-диаграмм состояний — описывать динамическое поведение системы через различные состояния и переходы объектов в ответ на события (Рисунок 18). Это позволяет понять, как объекты изменяют свое состояние в ходе выполнения процессов и какие события вызывают эти изменения.

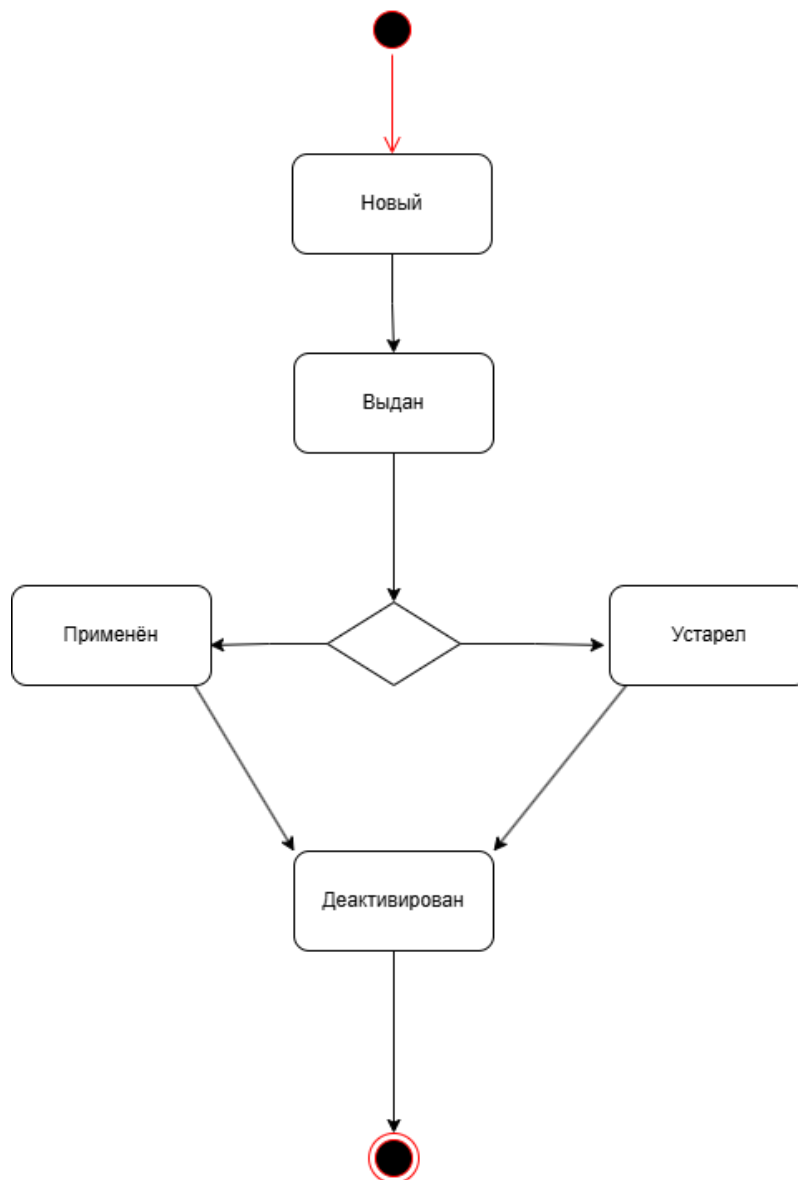


Рисунок 18. UML-диаграммы состояний

### 3.3 Сравнение систем-аналогов

**Gearbest.com** — это онлайн-магазин, предлагающий широкий ассортимент электроники и реплик.

#### Преимущества:

1. Четкая навигация: Удобный интерфейс для поиска и сортировки товаров.
2. Гарантия низкой цены: Постоянные акции и какие-либо предложения на популярные модели.
3. Лояльная программа наград: Возможность зарабатывать баллы за покупки, которые можно использовать для будущих заказов.

#### Недостатки:

1. Разное качество товаров: Реплики могут варьироваться по качеству от очень хороших до некачественных.
2. Долгий срок доставки: Занимает время, особенно при доставке в другие страны или даже города.

**DealExtreme.com** — интернет-магазин с широким ассортиментом, включая копии электроники от Apple.

**Преимущества:**

1. Множество категорий товаров: В наличии как реплики, так и аксессуары для устройств Apple.
2. Низкие цены: Частые скидки и предложения для экономии на покупке.
3. Отличный раздел отзывов: Возможность ознакомиться с мнением других пользователей, что помогает понять качество товара.

**Недостатки:**

1. Сложности с общением на языке: Большинство информации представлено на английском, но не всегда удобно для русскоязычных пользователей.
2. Задержки с доставкой: Сроки могут быть непредсказуемыми, особенно в пиковые сезоны.

**Banggood.com** — известный китайский интернет-магазин, который предлагает как оригинальную, так и реплицированную электронику.

**Преимущества:**

1. Регулярные акции: Частые распродажи и скидки на популярные модели реплик.
2. Подробные описания товаров: Каждая модель имеет детализированные характеристики и отзывы покупателей.
3. Доброжелательная поддержка: Команда поддержки помогает решать вопросы с покупателями.

**Недостатки:**

1. Не всегда точная информация о наличии товара: Бывают случаи, когда товар оказывается недоступен после оформления заказа.
2. Проблемы с возвратами: Условия возврата могут быть непонятны или усложнены для покупателей из других стран.

### **3.4 Экономическое обоснование разработки**

Затраты:

8. Разработка дизайна - 10000 рублей.
  9. Разработка кода - 12000 рублей.
  10. Тестирование - 3000 рублей.
  11. Хостинг и домен на первый год - 5000 рублей.
  12. Маркетинг и продвижение - 10000 рублей.
- Итого: 40000 рублей.

## ЗАКЛЮЧЕНИЕ

За время прохождения практики в студии интернет-решений «Grampus» (г. Вологда, ул. Мальцева, 52) мне удалось существенно расширить свой профессиональный кругозор. Практический опыт работы с современными инструментами разработки, такими как Visual Studio Code и Figma, стал ценным дополнением к теоретической базе, полученной за четыре года обучения в колледже.

Особую значимость представляет возможность применения академических знаний в реальных проектных условиях. Такой формат обучения демонстрирует высокую эффективность.

Не забыл и про технику безопасности (Приложение 1), уделив внимание защите данных пользователя и предотвращению распространенных уязвимостей. В результате разработанный сайт представляет собой функциональное и безопасное решение, соответствующее современным стандартам веб-разработки.

В заключение, хочу подчеркнуть, что прохождение практики в студии интернет-решений «Grampus» дало мне возможность значительно расширить мои навыки и знания. Я полностью удовлетворен этим процессом и благодарен команде студии за предоставленный мне ценный опыт. Я уверен, что все полученные знания и навыки будут служить мне в будущей профессиональной деятельности, помогая мне достигать успеха и преуспевать.

## Приложение 1

В процессе разработки сайта особое внимание было уделено вопросам информационной безопасности, что крайне важно в условиях современного интернета. Защита данных пользователей и предотвращение потенциальных угроз стали приоритетными задачами.

Таким образом, разработанный сайт включает в себя комплексный подход к безопасности, при этом все используемые компоненты и библиотеки регулярно обновляются для устранения известных уязвимостей. Однако важно помнить, что обеспечение безопасности — это непрерывный процесс, который требует постоянного мониторинга и адаптации к новым угрозам. В конечном итоге, реализованные меры значительно снижают риски и гарантируют пользователям высокий уровень защиты данных.