

«Вологодский колледж связи и информационных технологий»

Отчёт

О прохождении практики (учебная, производственная);

(Вид практики)

Студентом Ухановым Даниилом Андреевичем

Специальность «Информационные системы и программирование – разработчик мультимедийных приложений»

Курс 4 Форма обучения ОЧНАЯ Группа ИСП-421 Р

В Студии интернет-решений «Grampus»

(наименование организации)

Тема

с «14» октября 2024г. по «29» ноября 2024г.

Подпись студента _____
(подпись) (расшифровка)

« » 2024г.

Подпись руководителя

практики от колледжа _____ Зернова Н.В.
(подпись) (расшифровка)

« » 2024г.

Отчет принял

Преподаватель _____ Дусанюк Р.А.
(подпись) (расшифровка)

Оценка (прописью)

« » 2024

Вологда, 2024

СОДЕРЖАНИЕ

ВВЕДЕНИЕ.....	3
РАЗДЕЛ 1. ВЕРСТКА САЙТА	5
1.1. Этапы разработки сайта.....	5
1.2. Техническое задание	5
РАЗДЕЛ 2. ХОД ВЫПОЛНЕНИЯ ЗАДАНИЯ.....	7
2.1. Разработка HTML разметки.....	7
2.2. Стилизация сайта	8
2.3. Разработка логики сайта.....	9
РАЗДЕЛ 3. ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ	11
3.1 Разработка каркаса сайта.....	11
3.2 Разработка дизайна сайта	12
ЗАКЛЮЧЕНИЕ.....	17

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), было для нас 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).

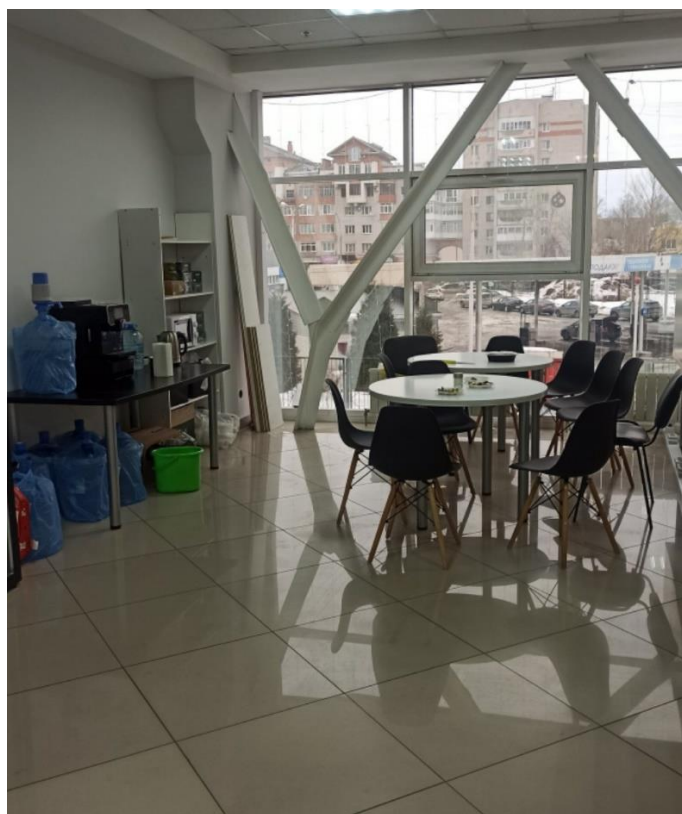


Рисунок 4. Grampus

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 `` 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

Основное задание:

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

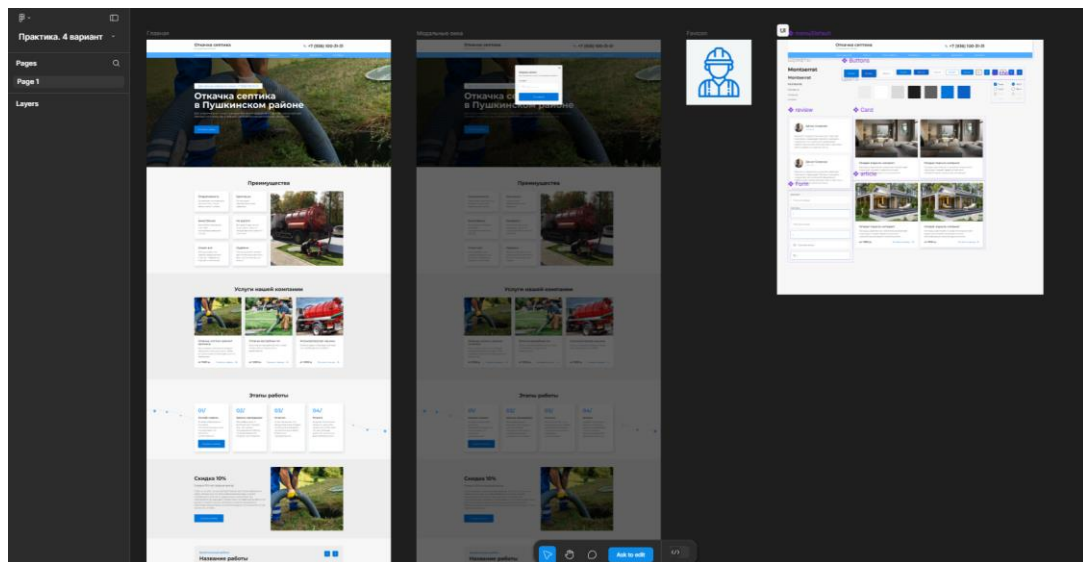


Рисунок 5. Макет сайта

Дополнительное задание:

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

РАЗДЕЛ 2. ХОД ВЫПОЛНЕНИЯ ЗАДАНИЯ

2.1. Разработка HTML разметки

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 6).
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>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 6. 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 **** 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 ****, 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 7) 2 секции **<div>**:

1. Лого компании, состоящее из 2 строк;
2. 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 7. 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.

В разрабатываемом сайте основной файл со стилями состоит из 800 строк, далее показаны основные стили для 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 8) указан атрибут «*font-family*» который устанавливает шрифты для всех элементов в значении «*Ubuntu*», который был указан в макете сайта и был импортирован из внешнего файла и подключен в начале файла со стилями.

```
* {
  padding: 0;
  margin: 0;

  color: #1D1D1B;
  font-family: Montserrat;

  --Primary: #0074D9;
  --Hover: #0A60C5;
  --Background: #F2F6F6;
  --Background2: #E9EDED;
  --Text: #606060;
  --Stroke: #D4D4D4;
  --Card: #FFFFFF;

  --MaxWidth: 1115px;
}
```

Рисунок 8. 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.

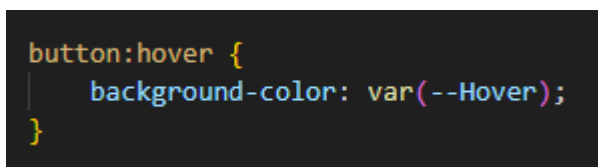
2.3. Разработка логики сайта

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.

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 9).

A screenshot of a code editor showing a CSS rule for a button's hover state. The code is:

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

Рисунок 9. 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», «PHP» 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.

Ссылка на скачивание сайта: <https://github.com/Nesar1337/GranpusUhan>

РАЗДЕЛ 3. ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ

3.1 Разработка каркаса сайта

Каркас сайта (или *wireframe*) представляет собой визуальное представление структуры веб-страницы (Рисунок 10), которое демонстрирует расположение основных элементов и функциональных зон.



Рисунок 10. Wireframe

1. Определение целей и задач:

Перед началом работы над каркасом сайта необходимо определить ключевые цели и задачи, которые должен решать веб-ресурс. Это могут быть такие аспекты, как привлечение пользователей, предоставление новостей (Рисунок 11), облегчение навигации и конверсии. На этой стадии также важным является анализ целевой аудитории и конкурсной среды, что позволит учесть предпочтения пользователей при создании интерфейса.



Рисунок 11. Новости

2. Сбор информации и идей:

Для создания эффективного каркаса полезно проанализировать существующие сайты в данной нише, изучить их структуру и функционал. Это поможет определить успешные решения и избегать распространённых ошибок.

3. Создание каркаса в Figma:

Figma — это мощный инструмент для разработки интерфейсов, который позволяет создавать высококачественные прототипы.

3.2 Разработка дизайна сайта

Я проанализировал примеры сайтов других клубов байкеров и разобрался что должно входить конструкцию сайта.

1. Определение концепции дизайна:

На начальном этапе разработки дизайна было важно определить концепцию, которая бы соответствовала культуре байкеров. Основные элементы, которые были учтены в концепции:

Эстетика: использование темных тонов (Рисунок 12), текстур, а также изображений, связанных с мотоциклами и активным образом жизни в духе свободной дороги.



Рисунок 12. Эстетика

Шрифты: выбор шрифтов с агрессивными, но читаемыми линиями, что обеспечило бы акцент на тематике байкерства.

Цветовая палитра: использование сочетаний черного, серого и ярких акцентов, таких как красный или оранжевый, благодаря чему визуально подчёркивалась динамика и энергия сообщества.

2. Создание макетов в Figma:

В процессе создания дизайна сайта в Figma были выполнены следующие шаги:

Дизайн главной страницы: Главная страница была разработана с акцентом на привлечение внимания и информативность. Включены следующие элементы:

Заголовок и логотип: размещение логотипа клуба в верхней части страницы (Рисунок 13), что создаёт первое визуальное впечатление.



Рисунок 13. Логотип

Секции с основным контентом: короткое приветственное сообщение о клубе (Рисунок 14).



Рисунок 14. «О нас»

Разработка внутренних страниц: Дизайн внутренних страниц, таких как «О нас», «Товары» и «Новости», был выполнен в едином стилевом решении (Рисунок 15).

О НАС ТОВАРЫ НОВОСТИ

Рисунок 15. Разделы

Страница «О нас»: включает лозунг, а также фотографию одного из байкеров на фоне его мотоцикла (Рисунок 16).

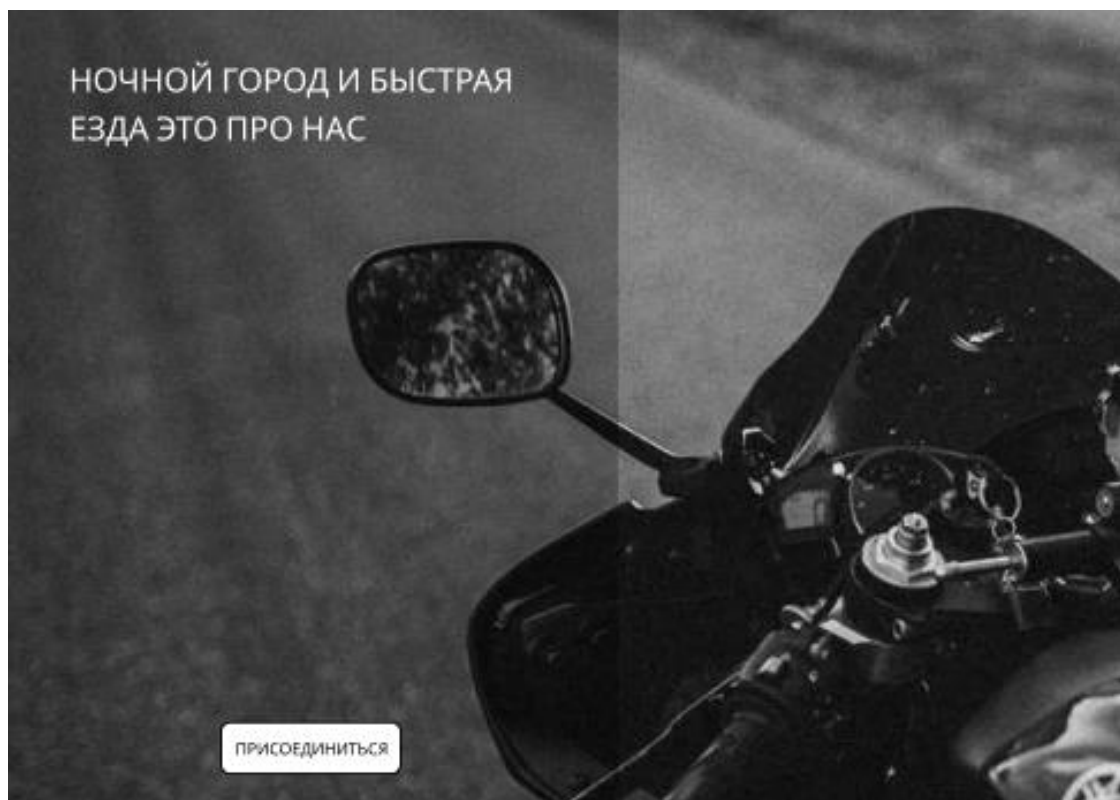


Рисунок 16. Лозунг

Страница «Товары»: витрина различных товаров начиная от курток, заканчивая фирменными наклейками (Рисунок 17).



Рисунок 17. Витрина товаров

Страница «Новости»: здесь еженедельно обновляются актуальные новости про мотомир и их клуб (Рисунок 18).

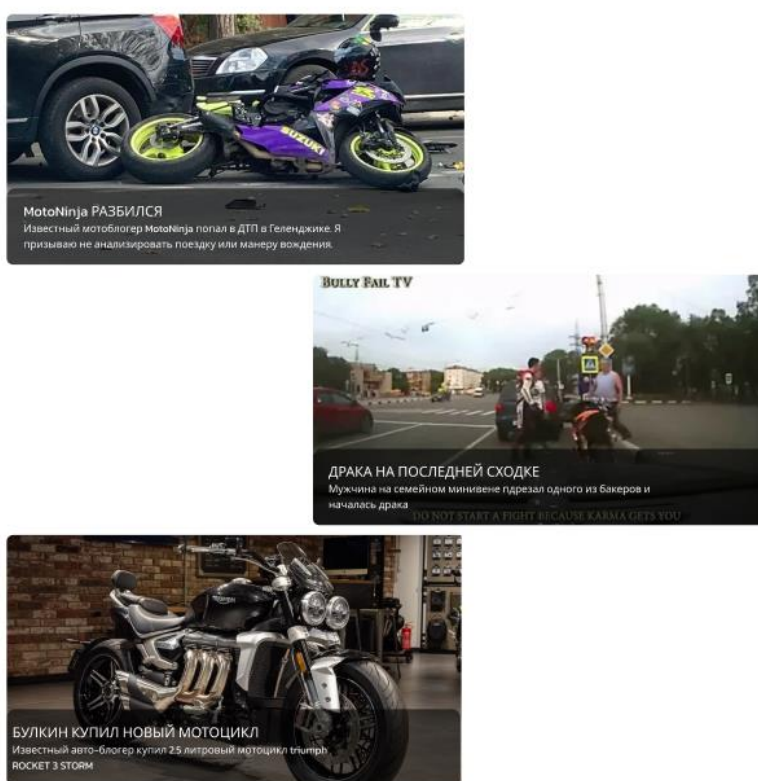


Рисунок 18. Новости

Интерактивные элементы. В процессе работы в Figma были добавлены интерактивные элементы, такие как кнопки (Рисунок 19) и выпадающие списки, что позволило создать прототип с функционалом, аналогичным готовому сайту.

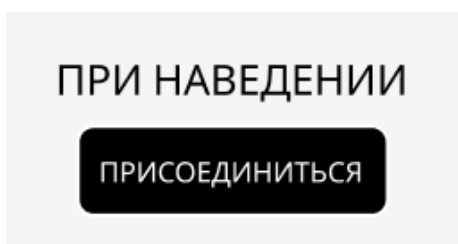


Рисунок 19. Интерактивные элементы

Подвал сайта

Подвал сайта является важным элементом, предоставляющим пользователям доступ к ключевой информации (Рисунок 20). В нем размещены ссылки на основные разделы, такие как «О нас», где размещен лозунг клуба, «Товары» тут размещены товары для приобретения, а также «Новости», чтобы узнать о произошедших случаях. Кроме того, в подвале присутствует кнопка возврата на начало страницы, расположенная по центру в низу, что позволяет удобно подняться к верхней части сайта. Контактные данные, включая электронную почту для обратной связи и номер телефона для прямого общения, находятся здесь же, добавляя удобство пользователям.

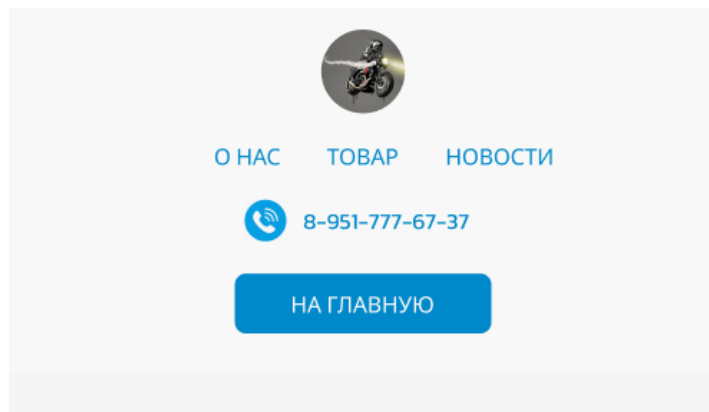


Рисунок 20. Подвал сайта

<https://github.com/Nesar1337/disinSaitaFigma>

ЗАКЛЮЧЕНИЕ

За ограниченное время, предоставленное для прохождения производственной практики в студии интернет-решений «Grampus» по адресу, Вологодская область, г. Вологда, ул. Мальцева, д. 52, 2 этаж, офис 208, я усвоил и закрепил целый ряд ценных навыков. В частности, я приобрел опыт в веб-верстке, научился работать с приложениями «Visual Studio Code» и «Figma». Я уверен, что все эти навыки окажут мне огромную помощь в моей дальнейшей карьере.

Весь этот опыт был для меня чрезвычайно полезным, особенно после трёх лет теоретического обучения в колледже, где мы получали базовые знания и концепции. Прохождение производственной практики помогло мне закрепить и применить эти теоретические знания на практике. Я считаю, что проведение практических занятий необходимо не только один раз, но как минимум дважды в рамках каждого учебного курса.

Однако, шесть недели, выделенные на производственную практику, оказались недолгим периодом времени. Я глубоко убежден в том, что для полного раскрытия потенциала и получения максимальной пользы от практического обучения, время для практики следует увеличить хотя бы вдвое. Это позволит нам получить еще больше практического опыта работы на реальных проектах и более глубоко освоить материал.

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.