**МИНИСТЕРСТВО ОБРАЗОВАНИЯ МОСКОВСКОЙ ОБЛАСТИ**

**Государственное бюджетное профессиональное образовательное  
учреждение Московской области  
«Авиационный техникум имени В.А. Казакова»**

ПЦК Профессионального цикла

Специальность: 09.02.07 Информационные системы и программирование

Квалификация: Программист

КУРСОВОЙ ПРОЕКТ (РАБОТА)

по междисциплинарному курсу  
МДК.01.01. Разработка программных модулей

(полное наименование МДК)

на тему: «Разработка»

Специальность 09.02.07 Информационные системы и программирование

Выполнил  
Обучающийся 3 курса

группы ИП-2 Иванов Данила Александрович+

преподаватель Кирилов Назари Андреевич

*(подпись)* (Фамилия Имя Отчество)

Курсовая работа защищена с оценкой

(оценка прописью)

г. Жуковский, 2024г.

**МИНИСТЕРСТВО ОБРАЗОВАНИЯ МОСКОВСКОЙ ОБЛАСТИ**

**Государственное бюджетное профессиональное образовательное  
учреждение Московской области  
«Авиационный техникум имени В.А. Казакова»**

ПЦК Профессионального цикла

Специальность 09.02.07 Информационные системы и программирование

Квалификация Программист

УТВЕРЖДАЮ «\_\_» 20\_\_ г.

/ /

(дата) (подпись)

**ЗАДАНИЕ НА ВЫПОЛНЕНИЕ КУРСОВОГО ПРОЕКТА (РАБОТЫ)  
по междисциплинарному курсу**

**МДК.01.01. Разработка программных модулей**

***Студентке(ту)* Кирилову Назари Андреевичу**

Специальность **09.02.07 Информационные системы и программирование**

Группа ИП-1 курс 3

1. Тема Разработка backend для веб-конференций
2. Дата выполнения курсового проекта (работы) «11» декобря 2024 г.
3. Содержание курсового проекта (работы):

**Цель:** Создание backend для проекта по веб-конференциям

преподаватель Громов Виталий Каприянович

*(подпись)* (Фамилия Имя Отчество)

*Задание принято к исполнению* «30» ноября 2024 г.

(подпись обучающегося)

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 4](#_Toc184132821)

[Глава 1. Теоретическая часть 5](#_Toc184132822)

[1.1. Хаб (hub) 5](#_Toc184132823)

[1.2. Контекст хаба 6](#_Toc184132824)

[1.3. Взаимодействие с клиентами 7](#_Toc184132825)

[1.4. Аутентификация и авторизация с помощью токенов 9](#_Toc184132826)

[1.5. Пользователи 10](#_Toc184132827)

[1.6. Группы 11](#_Toc184132828)

[Глава 2. Практическая часть 13](#_Toc184132829)

[1.1. Создание проекта 13](#_Toc184132830)

[1.2. Загрузка библиотеки 15](#_Toc184132831)

[1.3. Конфигурация приложения 19](#_Toc184132832)

[1.4. Конфигурация аутентификации и авторизации 19](#_Toc184132833)

[1.5. Создание хаба для подключения пользователей 21](#_Toc184132834)

# ВВЕДЕНИЕ

# В последние десятилетия веб-разработка стала одной из самых востребованных и динамично развивающихся областей информационных технологий. С каждым годом растет количество пользователей интернета, что создает огромный спрос на качественные веб-приложения и сайты. Веб-разработка охватывает широкий спектр задач, включая создание и поддержку веб-сайтов, разработку веб-приложений, а также интеграцию различных сервисов и технологий.

# Современные технологии веб-разработки позволяют создавать интерактивные и функциональные решения, которые могут удовлетворить потребности как бизнеса, так и конечных пользователей. Веб-разработка включает в себя как фронтенд (клиентская часть), так и бэкенд (серверная часть), что требует от разработчиков знаний в различных языках программирования, фреймворках и инструментах.

# Целью данной курсовой работы является изучение основных аспектов веб-разработки, включая технологии, используемые для создания веб-сайтов и приложений, а также процесс разработки от идеи до реализации. В рамках работы будут рассмотрены ключевые этапы разработки, такие как проектирование, программирование, тестирование и развертывание, а также лучшие практики и современные тренды в этой области.

# В первой главе будет представлен обзор технологий веб-разработки, включая HTML, CSS, JavaScript и серверные языки программирования. Во второй главе будет рассмотрен процесс проектирования и разработки веб-приложений, включая методы работы с базами данных и интеграцию API. В третьей главе будет проведен анализ современных инструментов и фреймворков, которые упрощают процесс разработки и повышают эффективность работы разработчиков.

# Таким образом, данная курсовая работа направлена на глубокое понимание веб-разработки как важной составляющей современного цифрового мира, а также на изучение практических аспектов создания веб-приложений, которые могут быть использованы в реальных проектах.

# Глава 1. Теоретическая часть

В начале расскажу немного про сам html. Мы разберем из чего состоит данное приложение, как его конфигурировать, что такое контекст хаба, как взаимодействовать с клиентами и многое другое.

## html (hub)

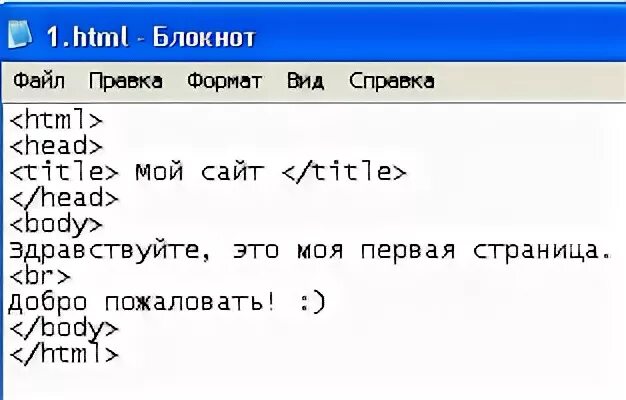
HTML (HyperText Markup Language) — это стандартный язык разметки, используемый для создания и структурирования содержимого веб-страниц. Он позволяет разработчикам описывать, как текст, изображения, видео и другие элементы должны отображаться в веб-браузере. HTML является основой для всех веб-сайтов и играет ключевую роль в веб-разработке. Основные функции HTML включают структурирование содержимого, создание гипертекстовых ссылок, встраивание медиа, создание форм для взаимодействия с пользователем и добавление метаданных. HTML позволяет организовывать текст и другие элементы на странице с помощью различных тегов, таких как заголовки, абзацы, списки и таблицы. Он также поддерживает создание ссылок, которые связывают разные страницы и ресурсы в интернете, что делает возможным навигацию между страницами. HTML позволяет встраивать изображения, видео и аудио на веб-страницы, создавая более интерактивные и привлекательные страницы. Кроме того, HTML предоставляет возможность создавать формы для ввода данных, которые могут быть использованы для сбора информации от пользователей. Знание HTML является основой для изучения других технологий веб-разработки, таких как CSS (для стилизации) и JavaScript (для добавления интерактивности). Правильная разметка HTML помогает улучшить видимость сайта в поисковых системах, что является важным аспектом цифрового маркетинга. HTML поддерживается всеми современными веб-браузерами, что обеспечивает кроссбраузерную совместимость и позволяет пользователям получать доступ к веб-сайтам на различных устройствах. В заключение, HTML — это фундаментальный язык для веб-разработки, который позволяет создавать и структурировать содержимое веб-страниц, необходимый для создания интерактивных, доступных и привлекательных веб-ресурсов.

Рисунок 1.1.1 – Простейший html

## польза css

CSS (Cascading Style Sheets) — это язык стилей, используемый для описания внешнего вида и форматирования HTML-документов. Он позволяет отделить содержание веб-страницы от ее представления, что делает разработку более гибкой и удобной. Вот основные аспекты, подчеркивающие пользу CSS:

Отделение содержания от оформления: CSS позволяет разделить структуру документа (HTML) и его визуальное представление. Это упрощает управление стилями и делает код более чистым и понятным.

Управление стилями: С помощью CSS можно легко изменять стили для множества элементов на странице. Например, изменение цвета текста или фона можно сделать в одном месте, и это изменение отразится на всех элементах, к которым применен этот стиль.

Адаптивный дизайн: CSS позволяет создавать адаптивные и отзывчивые веб-дизайны, которые хорошо выглядят на различных устройствах и экранах. С помощью медиа-запросов можно изменять стили в зависимости от размера экрана, что улучшает пользовательский опыт.

Улучшение визуального восприятия: CSS предоставляет множество возможностей для стилизации, включая шрифты, цвета, отступы, границы и тени. Это позволяет создавать более привлекательные и удобные для восприятия веб-страницы.

Кроссбраузерная совместимость: CSS помогает обеспечить единообразное отображение веб-страниц в различных браузерах. Хотя могут быть небольшие различия, использование стандартов CSS минимизирует проблемы с совместимостью.

Производительность: CSS может быть загружен один раз и кэширован браузером, что уменьшает время загрузки страниц при повторных посещениях. Это особенно важно для больших сайтов с множеством страниц.

Упрощение работы с макетом: CSS предоставляет различные методы для управления макетом, такие как Flexbox и Grid, которые позволяют легко создавать сложные и адаптивные структуры страниц.

Поддержка анимаций и переходов: CSS позволяет добавлять анимации и переходы к элементам, что делает интерфейс более интерактивным и привлекательным для пользователей.

Упрощение работы в команде: Разделение стилей и структуры позволяет командам разработчиков и дизайнеров работать более эффективно. Дизайнеры могут сосредоточиться на создании стилей, в то время как разработчики могут работать над функциональностью.

Улучшение SEO: Правильное использование CSS может улучшить SEO-оптимизацию сайта, так как он позволяет создавать более чистую и структурированную разметку, что облегчает индексацию страниц поисковыми системами. 

Рисунок 1.2.1 – Пример использования css контекста

## блочная верстка

Блочная верстка — это метод создания веб-страниц, основанный на использовании блоков, которые располагаются вертикально и могут содержать различные элементы, такие как текст, изображения и другие блоки. Основным элементом блочной верстки является тег <div>, который используется для группировки контента и создания структуры страницы.

Основные характеристики блочной верстки:

Структурирование контента: Блочная верстка позволяет организовать контент на странице в виде отдельных блоков, что упрощает его восприятие и редактирование.

Гибкость: Использование блоков дает возможность легко изменять расположение и размеры элементов на странице, что делает дизайн более адаптивным.

Стилизация: Блоки можно стилизовать с помощью CSS, задавая им различные свойства, такие как цвет, размер, отступы и границы.

Преимущества блочной верстки:

Упрощение кода: Блочная верстка позволяет избежать сложных таблиц и упрощает структуру HTML-кода.

Совместимость с современными стандартами: Блочная верстка соответствует современным стандартам веб-разработки, что делает сайты более доступными и удобными для пользователей.

Легкость в поддержке: Изменения в дизайне или структуре страницы можно вносить без значительных усилий, что упрощает процесс поддержки сайта.

Примеры использования:

Создание макетов страниц с несколькими колонками.

Разработка адаптивных дизайнов, которые хорошо отображаются на различных устройствах.

Использование фреймворков, таких как Bootstrap, для упрощения процесса блочной верстки.

Блочная верстка является основой современного веб-дизайна и широко используется для создания функциональных и эстетически привлекательных сайтов.



# Глава 2. Практическая часть

В данной части, мы будем писать web сайт.

## Создание проекта и его структура

Для начала необходимо создадать проект. Для создания и написания web мы будем использовать Visual Studio code .



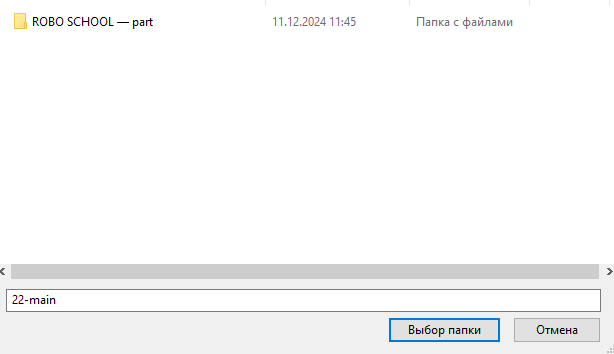


Рисунок 2.1.1 – Visual Studio code

Далее мы структерируем проэк

Папка css для стилизации img для фотографий

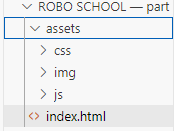
Js для функционала и index.html для основного 

Рисунок 2.1.2 – структура папок

В папки Css соделжится 2 файла normalize служит как ночальная иструкцыя Stayle для первостепенны

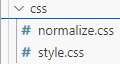
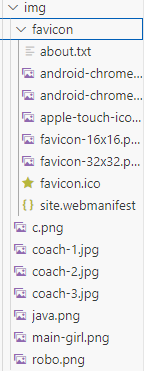


Рисунок 2.1.3 –файлы в папки css

Рисунок 2.1.4

В папки img в favicon мы содержим кортинки код для вывыдение иконки сайта

В оставшестве пространстве остальные фото которые будут использоваться в проэкте



В папки js содержится файл js



Рисунок 2.1.5 -файлы в папки js

файлы в папки img

## Начало разметки

В head мы добовляем название <title>Robo School</title> , шрифты

Под Fonts, иконку под Favicon и октеверуем стили под css



Рисунок 2.2.1-head

Header служит для новигации используя сылки

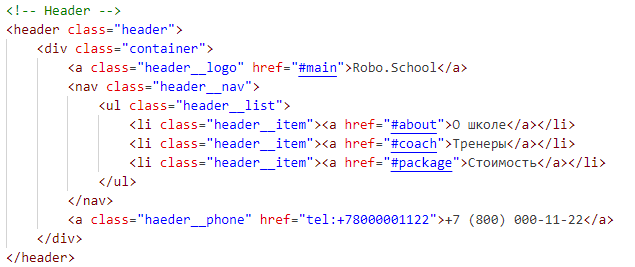


Рисунок 2.2.2 - Header

main служит как титульник

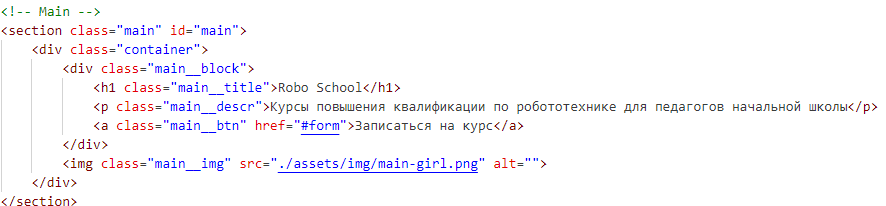


Рисунок 2.2.3 - Main

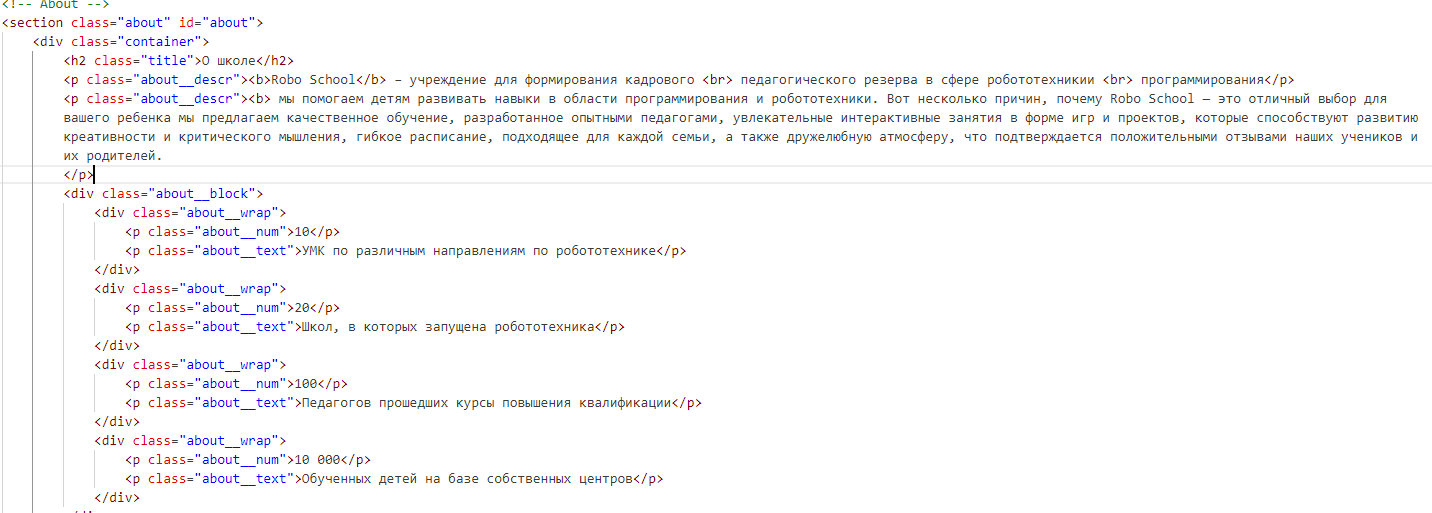
About содержит информаци про сеть школ robo shool

Рисунок 2.2.4 – About

Coach служит для расказа о персонале

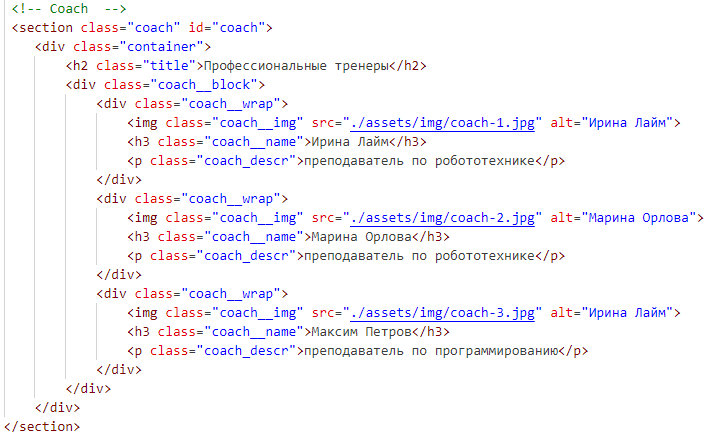


Рисунок 2.2.5 – Coach

Package служит чтобы расказать о продукте



Рисунок 2.2.6 – About

Form это обратная связь она позволяет заказать курс

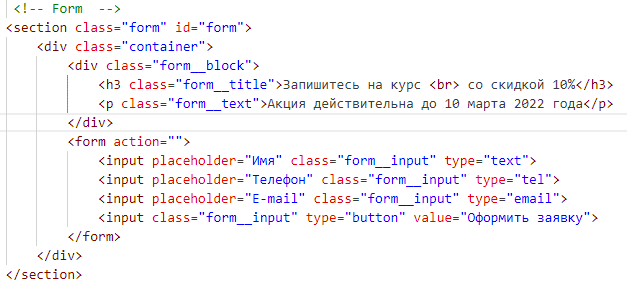


Рисунок 2.2.6 –From

## Css

Переменные

:root: определяет набор переменных, которые можно использовать в CSS.

--color-white: белый цвет, используемый в качестве фона или текста.

--color-dark: темный цвет, используемый в качестве текста или фона.

--color-zero: цвет, используемый для акцента или выделения.

--color-text: цвет текста, используемый по умолчанию.

--color-accent: цвет акцента, используемый для выделения важных элементов.

Глобальные стили

\*: устанавливает значение box-sizing для всех элементов в border-box, что позволяет включать padding и border в размер элемента.

html: устанавливает поведение прокрутки в smooth, что делает прокрутку более плавной.

body: устанавливает семейство шрифтов в 'Jost', размер шрифта в 18px и цвет текста в --color-dark.

a: удаляет подчеркивание у ссылок.

.container: устанавливает максимальную ширину контейнера в 1160px и центрирует его по горизонтали.

.title: устанавливает размер шрифта в 50px, вес шрифта в 600 и добавляет отступ снизу в 60px.

Примечания

Использование переменных позволяет легко менять цветовую схему сайта.

Установка box-sizing в border-box позволяет избежать проблем с размерами элементов.

Установка семейства шрифтов и размера шрифта в body позволяет установить общий стиль текста на сайте.

Установка поведения прокрутки в smooth делает прокрутку более плавной и комфортной для пользователей.

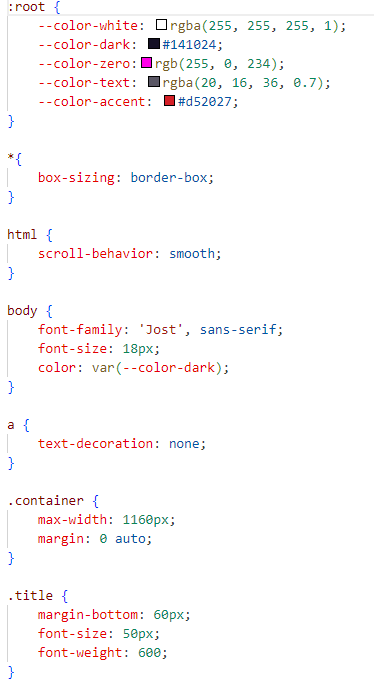


Рисунок 2.3.1 –общие

Заголовок

.header: устанавливает позицию заголовка в fixed, что позволяет ему оставаться на одном месте при прокрутке страницы. Устанавливает z-index в 1, что позволяет ему быть поверх других элементов. Устанавливает ширину в 100%, что позволяет ему занять всю ширину страницы. Устанавливает цвет фона в --color-dark и добавляет тень, что делает его более выразительным.

.header a: устанавливает цвет текста ссылок в --color-white и добавляет переход при наведении, что делает его более интерактивным.

.header a:hover: устанавливает цвет текста ссылок при наведении в --color-accent, что делает его более выразительным.

Контейнер заголовка

.header .container: устанавливает display в flex, что позволяет ему быть гибким и адаптивным. Устанавливает justify-content в space-between, что позволяет ему равномерно распределять элементы по горизонтали. Устанавливает align-items в center, что позволяет ему центрировать элементы по вертикали. Устанавливает высоту в 80px и максимальную ширину в 1500px, что позволяет ему быть достаточно большим для содержания элементов. Устанавливает padding в 0 20px, что позволяет ему иметь небольшой отступ с обеих сторон.

Логотип

.header\_\_logo: устанавливает размер шрифта в 30px, что делает его достаточно большим для логотипа. Устанавливает text-transform в uppercase, что делает его заглавным. Устанавливает font-weight в 600, что делает его более жирным. Устанавливает переход при наведении, что делает его более интерактивным.

Список

.header\_\_list: устанавливает display в flex, что позволяет ему быть гибким и адаптивным. Устанавливает align-items в center, что позволяет ему центрировать элементы по вертикали.

Примечания

Использование fixed позиции для заголовка позволяет ему оставаться на одном месте при прокрутке страницы.

Установка z-index в 1 позволяет ему быть поверх других элементов.

Использование тени и переходов при наведении делает заголовок более выразительным и интерактивным.

Использование flexbox для контейнера заголовка позволяет ему быть гибким и адаптивным.

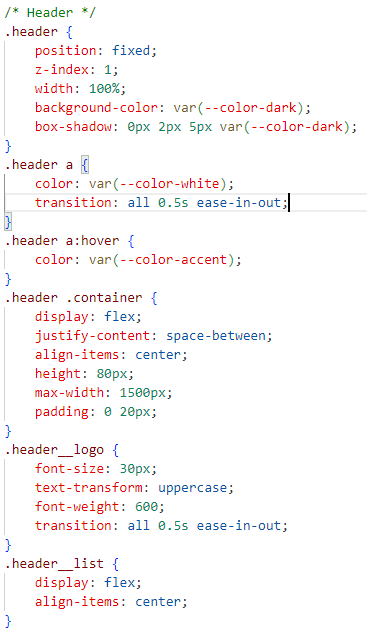


Рисунок 2.3.4 –.header

Подчеркивание ссылок

.header\_\_item a::after: добавляет подчеркивание под ссылками в элементах заголовка. Устанавливает content в пустую строку, что позволяет ему быть пустым элементом. Устанавливает display в block, что позволяет ему быть блоковым элементом. Устанавливает position в absolute, что позволяет ему быть абсолютным элементом. Устанавливает right в 0, что позволяет ему быть справа от ссылки. Устанавливает bottom в -3px, что позволяет ему быть ниже ссылки. Устанавливает width в 0, что позволяет ему быть невидимым. Устанавливает height в 2px, что позволяет ему быть тонким. Устанавливает background-color в --color-accent, что позволяет ему быть цветом акцента. Устанавливает transition в width 0.5s, что позволяет ему быть анимированным.

.header\_\_item a:hover::after: изменяет подчеркивание под ссылками при наведении. Устанавливает width в 100%, что позволяет ему быть видимым. Устанавливает left в 0, что позволяет ему быть слева от ссылки. Устанавливает background-color в red, что позволяет ему быть красным.

Телефон

.haeder\_\_phone: устанавливает размер шрифта телефона в 30px, что делает его достаточно большим.

Примечания

Использование ::after позволяет добавить подчеркивание под ссылками без изменения HTML-кода.

Установка position в absolute позволяет подчеркиванию быть абсолютным элементом.

Установка transition в width 0.5s позволяет подчеркиванию быть анимированным.

Использование background-color в --color-accent позволяет подчеркиванию быть цветом акцента.

Изменение подчеркивания при наведении позволяет сделать его более интерактивным.

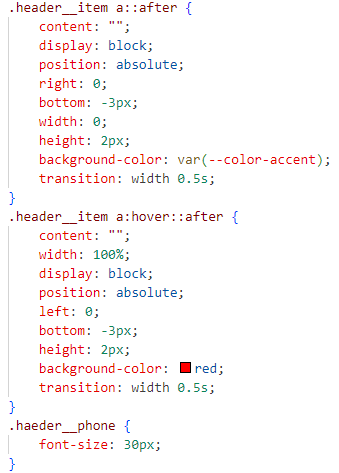


Рисунок 2.3.5 –.header\_\_item a:hover::after

Контейнер основного раздела

.main .container: устанавливает display в flex, что позволяет ему быть гибким и адаптивным. Устанавливает align-items в center, что позволяет ему центрировать элементы по вертикали. Устанавливает justify-content в space-between, что позволяет ему равномерно распределять элементы по горизонтали.

Заголовок основного раздела

.main\_\_title: устанавливает margin-bottom в 20px, что позволяет ему иметь отступ снизу. Устанавливает font-size в 70px, что делает его достаточно большим. Устанавливает text-transform в uppercase, что делает его заглавным. Устанавливает color в --color-white, что позволяет ему быть белым.

Описание основного раздела

.main\_\_descr: устанавливает font-size в 20px, что делает его достаточно большим. Устанавливает color в --color-white, что позволяет ему быть белым. Устанавливает margin-bottom в 50px, что позволяет ему иметь отступ снизу. Устанавливает line-height в 160%, что позволяет ему иметь более комфортный интервал между строками.

Кнопка основного раздела

.main\_\_btn: устанавливает display в inline-block, что позволяет ему быть блоковым элементом. Устанавливает background-color в --color-accent, что позволяет ему быть цветом акцента. Устанавливает color в --color-white, что позволяет ему быть белым. Устанавливает padding в 25px 55px, что позволяет ему иметь отступы. Устанавливает border в 2px solid --color-accent, что позволяет ему иметь границу. Устанавливает border-radius в 5px, что позволяет ему иметь закругленные углы. Устанавливает transition в all 0.5s ease-in-out, что позволяет ему быть анимированным.

.main\_\_btn:hover: изменяет кнопку при наведении. Устанавливает background-color в --color-dark, что позволяет ему быть темным. Устанавливает color в --color-accent, что позволяет ему быть цветом акцента.

Изображение основного раздела

.main\_\_img: устанавливает max-width в 622px, что позволяет ему иметь максимальную ширину.



Рисунок 2.3.6 –main

Раздел "О нас"

.about: устанавливает padding в 120px 0, что позволяет ему иметь отступы сверху и снизу.

.about\_\_descr: устанавливает font-size в 30px, что делает его достаточно большим. Устанавливает margin-bottom в 60px, что позволяет ему иметь отступ снизу.

.about\_\_block: устанавливает display в flex, что позволяет ему быть гибким и адаптивным. Устанавливает justify-content в space-between, что позволяет ему равномерно распределять элементы по горизонтали. Устанавливает flex-wrap в wrap, что позволяет ему переносить элементы на новую строку.

.about\_\_wrap: устанавливает width в 24%, что позволяет ему иметь ширину 24% от родительского элемента.

.about\_\_num: устанавливает font-size в 70px, что делает его достаточно большим. Устанавливает color в --color-accent, что позволяет ему быть цветом акцента. Устанавливает font-weight в 600, что делает его более жирным. Устанавливает margin-bottom в 20px, что позволяет ему иметь отступ снизу.

.about\_\_text: устанавливает color в --color-text, что позволяет ему быть цветом текста.



Рисунок 2.3.6 - About

Раздел "Тренеры"

.coach: устанавливает padding в 120px 0, что позволяет ему иметь отступы сверху и снизу.

.coach\_\_block: устанавливает display в flex, что позволяет ему быть гибким и адаптивным. Устанавливает justify-content в space-between, что позволяет ему равномерно распределять элементы по горизонтали.

.coach\_\_img: устанавливает margin-bottom в 20px, что позволяет ему иметь отступ снизу. Устанавливает width в 350px, что позволяет ему иметь ширину 350px. Устанавливает transition в 0.5s all ease-in-out, что позволяет ему быть анимированным. Устанавливает border-radius в 20px, что позволяет ему иметь закругленные углы.

.coach\_\_img:hover: изменяет изображение при наведении. Устанавливает transform в scale(1.05), что позволяет ему быть увеличенным на 5%.

.coach\_\_name: устанавливает font-size в 24px, что делает его достаточно большим. Устанавливает margin-bottom в 2px, что позволяет ему иметь отступ снизу.



Рисунок 2.3.6 – Coach

.card-container: устанавливает display в flex, что позволяет ему быть гибким и адаптивным. Устанавливает gap в 20px, что позволяет ему иметь отступы между карточками. Устанавливает margin-right в 35px, что позволяет ему иметь отступ справа.

Карточка

.card: устанавливает width в 350px, что позволяет ему иметь ширину 350px. Устанавливает height в 400px, что позволяет ему иметь высоту 400px. Устанавливает margin в 20px, что позволяет ему иметь отступы. Устанавливает border-radius в 20px, что позволяет ему иметь закругленные углы. Устанавливает perspective в 1000px, что позволяет ему иметь перспективу.

Внутренний контент карточки

.card-inner: устанавливает position в relative, что позволяет ему быть относительным элементом. Устанавливает width и height в 100%, что позволяет ему иметь ширину и высоту 100% от родительского элемента. Устанавливает transition в transform 0.6s, что позволяет ему быть анимированным. Устанавливает transform-style в preserve-3d, что позволяет ему сохранять 3D-перспективу.

Анимация карточки

.card:hover .card-inner: изменяет внутренний контент карточки при наведении. Устанавливает transform в rotateY(180deg), что позволяет ему быть повернутым на 180 градусов.

Заголовок карточки

.card-h: устанавливает margin в 10px, что позволяет ему иметь отступы.

Передняя и задняя стороны карточки

.card-front, .card-back: устанавливает position в absolute, что позволяет ему быть абсолютным элементом. Устанавливает width и height в 100%, что позволяет ему иметь ширину и высоту 100% от родительского элемента. Устанавливает backface-visibility в hidden, что позволяет ему быть невидимым при повороте. Устанавливает display в flex, что позволяет ему быть гибким и адаптивным. Устанавливает justify-content в center, что позволяет ему центрировать содержимое по горизонтали. Устанавливает align-items в center, что позволяет ему центрировать содержимое по вертикали. Устанавливает font-size в 24px, что делает его достаточно большим. Устанавливает color в white, что позволяет ему быть белым.

Передняя сторона карточки

.card-front: устанавливает background-color в #007bff, что позволяет ему быть синим. Устанавливает border-radius в 20px, что позволяет ему иметь закругленные углы.

Задняя сторона карточки

.card-back: устанавливает background-color в #dc3545, что позволяет ему быть красным. Устанавливает transform в rotateY(180deg), что позволяет ему быть повернутым на 180 градусов. Устанавливает border-radius в 20px, что позволяет ему иметь закругленные углы.Изображения на карточках 

Рисунок 2.3.6 – card

#robo, #programc, #java: устанавливают background-image в соответствующее изображение, что позволяет ему иметь изображение на карточке. Устанавливает margin в 0, что позволяет ему иметь отступы. Устанавливает background-size в cover, что позволяет ему заполнять весь экран. Устанавливает background-position в center, что позволяет ему центрировать изображение.

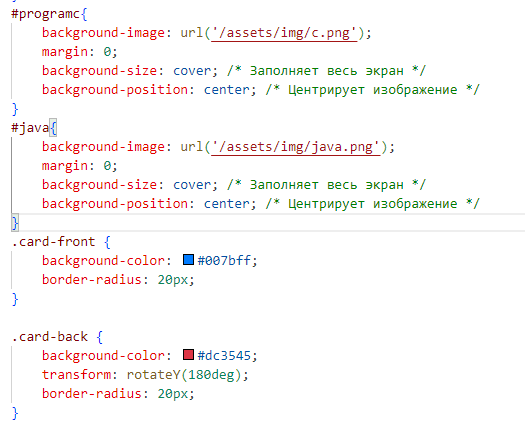


Рисунок 2.3.6 – card

Эти стили определяют вид и поведение формы на сайте.

Форма

.form: устанавливает background-color в var(--color-accent), что позволяет ей быть цветом акцента. Устанавливает padding в 90px 0, что позволяет ей иметь отступы сверху и снизу. Устанавливает color в var(--color-white), что позволяет ей быть белой.

Контейнер формы

.form .container: устанавливает max-width в 900px, что позволяет ей иметь максимальную ширину 900px. Устанавливает display в flex, что позволяет ей быть гибкой и адаптивной. Устанавливает justify-content в space-between, что позволяет ей равномерно распределять элементы по горизонтали. Устанавливает align-items в center, что позволяет ей центрировать элементы по вертикали.

Заголовок формы

.form\_\_title: устанавливает font-size в 30px, что делает ее достаточно большой. Устанавливает line-height в 130%, что позволяет ей иметь более комфортный интервал между строками. Устанавливает margin-bottom в 20px, что позволяет ей иметь отступ снизу.

Поле формы

.form\_\_input: устанавливает display в block, что позволяет ей быть блоковым элементом. Устанавливает width в 500px, что позволяет ей иметь ширину 500px. Устанавливает margin-bottom в 10px, что позволяет ей иметь отступ снизу. Устанавливает border-radius в 5px, что позволяет ей иметь закругленные углы. Устанавливает border в none, что позволяет ей не иметь границы. Устанавливает padding в 15px, что позволяет ей иметь отступы.

Кнопка формы

.form\_\_input[type="button"]: устанавливает background-color в var(--color-dark), что позволяет ей быть темной. Устанавливает color в var(--color-white), что позволяет ей быть белой. Устанавливает cursor в pointer, что позволяет ей иметь указатель мыши. Устанавливает border в 2px solid var(--color-dark), что позволяет ей иметь границу. Устанавливает transition в all 0.5s ease-in-out, что позволяет ей быть анимированной.

.form\_\_input[type="button"]:hover: изменяет кнопку при наведении. Устанавливает background-color в transparent, что позволяет ей быть прозрачной.



Рисунок 2.3.6 – form

Эти медиа-запросы определяют стили для различных размеров экрана.

Экраны с шириной до 1199px

..container: устанавливает max-width в 960px, что позволяет ему иметь максимальную ширину 960px.

.about\_\_num: устанавливает font-size в 60px, что делает его меньше.

.main\_\_img: устанавливает max-width в 311px, что позволяет ему иметь максимальную ширину 311px.

.main\_\_title: устанавливает margin-bottom в 10px, что позволяет ему иметь отступ снизу. Устанавливает font-size в 35px, что делает его меньше.

Экраны с шириной до 991px

.container: устанавливает max-width в 720px, что позволяет ему иметь максимальную ширину 720px.

.about\_\_wrap: устанавливает width в 48%, что позволяет ему иметь ширину 48% от родительского элемента. Устанавливает margin-top в 20px, что позволяет ему иметь отступ сверху.

.header\_\_logo: устанавливает font-size в 15px, что делает его меньше.

.header\_\_phone: устанавливает font-size в 15px, что делает его меньше.

.header\_\_item: устанавливает margin в 0 10px, что позволяет ему иметь отступы.

Экраны с шириной до 767px

.container: устанавливает max-width в 540px, что позволяет ему иметь максимальную ширину 540px.

Экраны с шириной до 575px

.container: устанавливает max-width в 96%, что позволяет ему иметь максимальную ширину 96% от родительского элемента.

.about\_\_wrap: устанавливает width в 100%, что позволяет ему иметь ширину 100% от родительского элемента. Устанавливает margin-top в 20px, что позволяет ему иметь отступ сверху.

Эти медиа-запросы позволяют адаптировать стили для различных размеров экрана, что делает сайт более удобным для пользователей с разными устройствами.

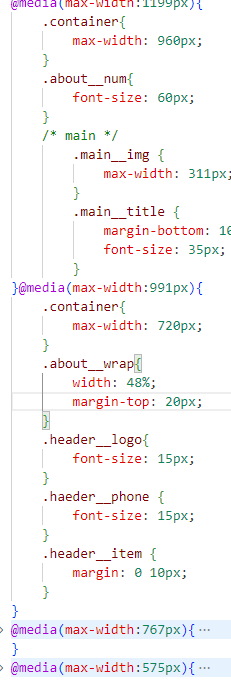


Рисунок 2.3.6 – media

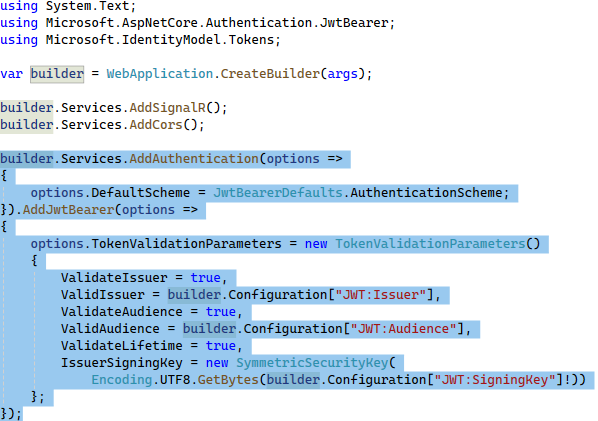
## Конфигурация аутентификации и авторизации

Перед настройкой приложения, необходимо добавить значения для настройки JWT-токенов.



Рисунок 2.4.1 – Добавления конфигурации для настройки JWT

Следующим шагом нам необходимо настроить и добавить в middleware аутентификации и авторизации(Рисунок 2.4.2).



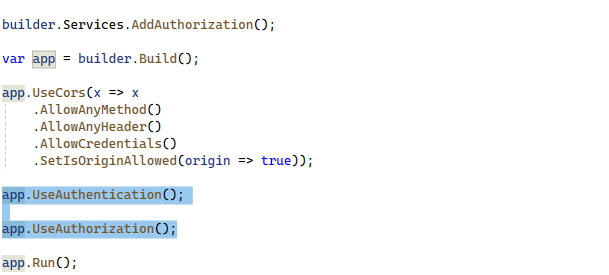


Рисунок 2.4.2 – Конфигурация JWT в проекте

## Создание хаба для подключения пользователей

Следующим нашим шагом будет создание хаба. Про сам хаб уже было все рассказано, здесь будет лишь показана практика и какие задачи он должен решать.

Вначале в решении проекта создается папка под название «Hubs». Вы можете называть папку как вам угодно, но в сообществе рекомендуют называть ее как «Hubs». Затем мы создаем файл, который как раз таки будет являться хабом(Рисунок 2.5.1).

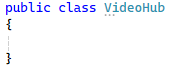


Рисунок 2.5.1 – Пустой хаб

Далее необходимо унаследовать данный класс от класса hub, для того чтобы система смогла понять, что данный класс является хабом, а также для того чтобы мы в принципе могли написать методы хаба.

Вначале для передачи данных с клиента, необходимо определить классы необходимые при работе с клиентами. В ней указаны данные которые мы будет принимать от клиента и данные которые будут отправлены клиентам группы(Рисунок 2.5.2).

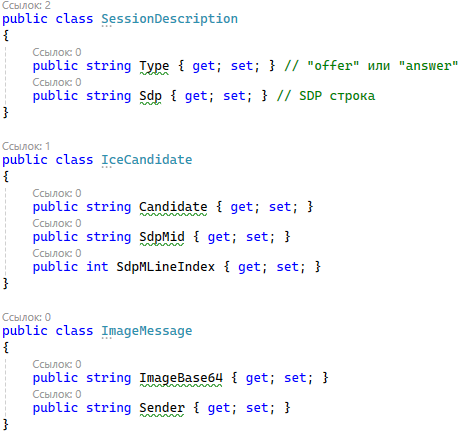


Рисунок 2.5.2 – Создание классов для работы с данными клиента

Далее необходимо создать сами методы(Рисунок 2.5.3).



Рисунок 2.5.3 – Создание методов для работы с клиентами

На данном рисунке указаны методы, через которые клиент может создать конференцию, подключиться к конференции, а также отправлять видеопоток данных для трансляции видео.

Ну и последним шагом является регистрация хаба в middleware, для того чтобы клиенты смогли к нему подключиться(Рисунок 2.5.4).



Рисунок 2.5.4 – Регистрация хаба в middleware

# ЗАКЛЮЧЕНИЕ

В заключение, разработка приложений для веб-конференций представляет собой важный шаг в эволюции коммуникационных технологий, особенно в условиях современного мира, где удаленная работа и глобализация становятся нормой. Использование таких технологий, как SignalR, позволяет обеспечить надежную и эффективную связь между участниками конференции, что значительно улучшает качество взаимодействия и обмена информацией. В данной работе мы подробно рассмотрели процесс создания backend-части приложения для веб-конференций, что не только углубляет понимание технических аспектов разработки, но и предоставляет читателям необходимые знания для создания собственных решений в этой области. Таким образом, данная работа служит не только теоретическим, но и практическим руководством для всех, кто стремится освоить создание современных коммуникационных приложений.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

Microsoft Learn - https://learn.microsoft.com/ru-ru/aspnet/core/tutorials

Microsoft Docs - https://learn.microsoft.com/ru-ru/aspnet/core/signalr

GitHub - https://github.com/dotnet/aspnetcore/tree/main/src/SignalR/samples

Pluralsight - https://www.pluralsight.com/courses/aspnet-core-signalr

Udemy - https://www.udemy.com/topic/signalr

YouTube - https://www.youtube.com

Stack Overflow - https://stackoverflow.com/questions/tagged/signalr

Dev.to - https://dev.to/t/signalr

Medium - https://medium.com/tag/signalr

C# Corner - https://www.c-sharpcorner.com/search/?q=signalr

DotNetCurry - https://www.dotnetcurry.com/tag/signalr

Telerik - https://www.telerik.com/blogs/tag/signalr

InfoWorld - https://www.infoworld.com/category/signalr

ASP.NET Blog - https://devblogs.microsoft.com/aspnet/category/signalr

Microsoft Q&A - https://docs.microsoft.com/answers/topics/signalr.html

ASP.NET SignalR Documentation - https://learn.microsoft.com/en-us/aspnet/signalr

Microsoft Azure SignalR Service - https://learn.microsoft.com/en-us/azure/azure-signalr

SignalR with Angular - https://www.c-sharpcorner.com/article/signalr-with-angular

SignalR with React - https://www.c-sharpcorner.com/article/signalr-with-react

SignalR with Vue.js - https://www.c-sharpcorner.com/article/signalr-with-vue-js

SignalR and Blazor - https://www.telerik.com/blogs/signalr-and-blazor

SignalR Security Best Practices - https://www.c-sharpcorner.com/article/signalr-security-best-practices

SignalR and WebSockets - https://www.telerik.com/blogs/signalr-and-websockets