ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

«ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Д09.03.03-ИНФ-19-19/3779.КП

**Кафедра**искусственного интеллекта

и системного анализа

КУРСОВОЙ ПРОЕКТ

по дисциплине: «Web-технологии»

Тема: «Разработка web-сайта клуба.»

Руководители:

(дата, подпись) ст. преп.Е.А. Маслова

Нормоконтроль:

(дата, подпись) ст. преп.Е.Н. Едемская

Разработал:

(дата, подпись) ст. гр. ИНФ-19 А.А. Лапин

Донецк – 2022

**Государственное ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ   
ВЫСШЕГО ПРОВЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ**

**«Донецкий национальный технический университет»**

**Факультет:** ***Компьютерных наук и технологий***

**Направление*: Прикладная информатика***

**Профиль: *Прикладная информатика***

**Кафедра:** «***Искусственный интеллект и системный анализ»***

ЗАДАНИЕ НА КУРСОВОЙ ПРОЕКТ

по дисциплине «Операционные системы»

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Студенту | Лапину Артуру Анатольевичу | | | | | | группы | ИНФ-19 |
|  | (фамилия, имя, отчество) | | | | | |  |  |
|  |  | | | | | |  |  |
| Тема проекта | Разработка web-сайта клуба. | | | | | | | |
|  | | | | | | | | |
| Исходные данные к проекту | | | | разработка web-сайта в соответствии с | | | | |
| заданием; | | | |  | | | | |
| оформление курсовой работы в соответствии с заданными требованиями; | | | | | | | | |
| подготовка и защита проекта. | | | | | | | | |
|  | | | | | | | | |
|  | | | | | | | | |
|  | | | | | | | | |
|  | | | | | | | | |
| Рекомендуемая литература | | | Таненбаум Э. Современные операционные | | | | | |
| системы / Э.Таненбаум 3-е изд. –СПб.: Питер, 2010. – 1120 с.: ил. | | | | | | | | |
|  | | | | | | | | |
| Дата выдачи задания | | 15.02.2022 | | |  |  | | |
|  | |  | | |  |  | | |
| Дата защиты проекта | | 31.05.2022 | | |  |  | | |
|  | |  | | |  |  | | |
| **Руководитель** | |  | | |  | ст. преп. Е.А. Маслова | | |
|  | | (подпись) | | |  | (должность, Ф.И.О.) | | |
|  | |  | | |  | ст. преп. Е.Н. Едемская | | |
|  | | (подпись) | | |  | (должность, Ф.И.О.) | | |
| **Разработчик** | |  | | |  | А.А. Лапин | | |
|  | | (подпись) | | |  | (Ф.И.О) | | |

РЕФЕРАТ

Пояснительная записка: 29 с., 11 рис., 8 источников.

Целью выполнения курсовой работы по дисциплине Web-технологии является:

1) формирование умений:

* применять программные средства редактирования и просмотра web-страниц;
* проектировать структуру web-сайта;
* создавать web-страницы средствами языка HTML;
* использовать средства CSS, JavaScript;
* создавать интерактивные web-приложения.

2) формирование профессиональных компетенций:

* осуществлять сбор и анализ информации для определения потребностей клиента;
* разрабатывать и публиковать программное обеспечение и информационные ресурсы отраслевой направленности со статическим и динамическим контентом на основе готовых спецификаций и стандартов;
* проводить отладку и тестирование программного обеспечения отраслевой направленности.

3) формирование общих компетенций:

* понимать сущность и социальную значимость своей будущей профессии, проявлять к ней устойчивый интерес;
* организовывать собственную деятельность, выбирать типовые методы и способы выполнения профессиональных задач, оценивать их эффективность и качество;
* принимать решения в стандартных и нестандартных ситуациях и нести за них ответственность;
* осуществлять поиск и использование информации, необходимой для эффективного выполнения профессиональных задач, профессионального и личностного развития;
* использовать информационно-коммуникационные технологии в профессиональной деятельности.

HTML. CSS, JAVASCRIPT, ВЕРСТКА, ОБЪЕКТ, МОДУЛЬ, ТИПЫ, КЛАССЫ, ПРИЛОЖЕНИЕ, ИНФОРМАЦИЯ, ДАННЫЕ, ВЕБ-САЙТ.

СОДЕРЖАНИЕ

[**ВВЕДЕНИЕ** 6](#_Toc107085320)

[1 ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ 8](#_Toc107085321)

[1.1 Основные этапы создания веб-сайта 8](#_Toc107085322)

[1.1.1 Постановка целей и задач сайта 8](#_Toc107085323)

[1.1.2 Создание и проработка технического задание 8](#_Toc107085324)

[1.1.3 Создание макета дизайна сайта 9](#_Toc107085325)

[1.1.4 Верстка 9](#_Toc107085326)

[1.1.5 Программирование 10](#_Toc107085327)

[1.1.6 Наполнение контентом 10](#_Toc107085328)

[1.1.7 Тестирование 10](#_Toc107085329)

[1.2 Средства разработки 11](#_Toc107085330)

[1.2.1 HTML 11](#_Toc107085331)

[1.2.2 CSS 11](#_Toc107085332)

[1.2.3 JavaScript и API 11](#_Toc107085333)

[2 ПРАКТИЧЕСКА ЧАСТЬ 13](#_Toc107085334)

[2.1 Выбор варианта дизайна разрабатываемой информационной системы. 13](#_Toc107085335)

[2.2 Анализ структуры дизайна. 13](#_Toc107085336)

[2.4 Подключение и работа с CSS 16](#_Toc107085337)

[2.5 Наполнение всех страниц контентом 17](#_Toc107085338)

[2.6 Java Script’s 19](#_Toc107085339)

[ВЫВОДЫ 23](#_Toc107085340)

[ПЕРЕЧЕНЬ ИСПОЛЬЗОВАНЫХ ИСТОЧНИКОВ 24](#_Toc107085341)

[Приложение Б ЛИСТИНГИ 25](#_Toc107085342)

[Б.1 – Подключение google fonts и .css файла. .html, style.css 25](#_Toc107085343)

[Б.2 – Реализация горизонтального меню на сайте style.css 25](#_Toc107085344)

[Б.3 – Реализация тела страницы style.css 25](#_Toc107085345)

[Б.4 – Реализация подвала style.css 25](#_Toc107085346)

[Б.5 – Листинг java.js 26](#_Toc107085347)

[Б.6 – Листинг кнопки «Свяжитесь с нами» 26](#_Toc107085348)

# **ВВЕДЕНИЕ**

На сегодняшний день почти каждый человек в мире имеет доступ к сети интернет у себя дома или на работе. Согласно сведениям, одной из крупнейших в мире немецких исследовательских компаний «Общество по изучению потребителя», общая численность интернет-пользователей в возрасте старше 16 лет в 2016 г. составила, только в России 84 мл. человек, т.е. 70,4% жителей страны. Значительно выросла аудитория пользователей мобильного интернета – им пользуется 56 млн. граждан России (46,6 % от общего числа интернет-пользователей).

Все больше и чаще сегодня люди ищут и покупают товары и оплачивают услуги через интернет. Стремительно развивается так называемый офлайн и онлайн-шоппинг. Процесс покупки становится комфортным: вместо изматывающий походов по торговым центрам и магазинам человек выбирает товары, не выходя из дома, экономит силы и средства, при отсутствии очередей и большого скопления людей. Открывается возможность приобретать те товары, которые не производятся в вашей стране.

Наличие огромного числа пользователей и потенциальных клиентов создает основу для широкой коммерческой деятельности частных компаний. Современные технологии предлагают такие возможности для развития бизнеса, как электронная торговля через интернет-магазины, осуществление рекламных и брендинговых кампаний с минимальными вложениями, анализ рынков, современных трендов, спроса, вкусов и интересов покупателей и др.

Любое современное предприятие или организация нуждаются в грамотном веб-сайте, так как от этого зависит ее успех. К примеру, на собственном сайте компания может поместить информацию о проведении тендеров, семинаров или конференций с подробным описанием целей, задач, сроков, условий и т.д Все заинтересованные лица и организации смогут отслеживать эту информацию из любого географического пункта. Сайт предоставляет широкие возможности для поиска деловых партнеров, решения кадровых вопросов. Можно подбирать не только персонал, но и клиентов путем воздействия на особую, узкопрофильную, целевую аудиторию, к которой организация имеет интерес.

**Цель создания курсовой работы:** разработка веб-сайта для клуба

**Задачи:**

* 1. Выявить особенности контента и дизайна современных веб-сайтов;
  2. Выполнить обзор веб-сайтов тематики – вечернего клуба;
  3. Создать веб-страницы средствами языка HTML;
  4. Использовать средства CSS, JavaScript для создания интерактивных веб-приложений.

# 1 ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

## 1.1 Основные этапы создания веб-сайта

Последовательность создания сайта и четкая проработка этапов – залог успеха всего проекта. Каждый этап создания сайта – это кропотливая работа, за которую ответственен каждый разработчик из команды веб-студии.

Создание веб-сайта заключает в себе не только разработку дизайна и программинг, а и детальный анализ проекта, сотрудничество с заказчиком и поиск решений для достижения поставленных целей проекта.

### 1.1.1 Постановка целей и задач сайта

Далеко не каждый человек, который хочет себе сайт, может сказать о различиях веб-сайтов, разных видах и т.д. Так же он не сможет самостоятельно определить цели и задачи сайта напрямую.

Если человеку необходим веб-сайт для своего бизнеса, необходимо провести подготовительный этап – проанализировать для чего нужен сай, какие задачи он должен решать: выполнять функцию представительства компании в Интернете или же продавать товар и услуги, стать качественной рекламной компании или быть стартапом для молодой фирмы.

После того, как поставленызадачи, необходимо определить целевую аудиторию. Важно понять для кого создается ресурс. На основе полученных данных формируется структура сайта, контент, дизайн, функционал и т.д.

Далее проводится маркетинговый анализ, в который включен анализ конкурентов, сезонности товара или услуги, деятельности и ниши, конкурентоспособности предложения. Определяются трендовые направления и новейший функционал, который может предложить конкурент.

### 1.1.2 Создание и проработка технического задание

Техническое задание – это документ, который составляется вместе с заказчиком. Он включает детальные характеристики ресурса:

* Пожелания по дизайну (цветовое оформление, применение фирменного стиля или его разработка, соотношение графических элементов к текстовым);
* Семантическое ядро;
* Структуру сайта, количество страниц, категорий, блоков;
* Функционал сайта (только стандартный или разработка дополнительных модулей, подробное их описание и цели);
* Применяемые технологии;
* Технические требования к ресурсу.

### 1.1.3 Создание макета дизайна сайта

Все этапы создания сайта важны для качественного функционирования ресурса, но разработка дизайна – один из главных. Ведь дизайн – это то, что видит посетитель в первую очередь, оценивает его и принимает решение остаться на странице или закрыть вкладку браузера.

Дизайнер основывается на техническом задании, рисует кнопки, баннеры и другие графические элементы. Другими словами, тот прототип, который был создан на первом этапе разработки сайта, получает эстетичный внешний вид, производится в цветах, выбранными заказчиком. Если компания имеет корпоративный стиль, то дизайн разрабатывается в его соответствии. Или же может сначала разработаться фирменный стиль, а после - дизайн на его основе.

### 1.1.4 Верстка

Верстальщик с помощью языка HTML переводит готовый дизайн в рабочий проект. Ресурс получает жизнь, становится динамичным, все кнопки работают. Ресурс становится кроссбраузерным и правильно отображается во всех существующих интернет-браузерах. На это же этапе создаются стили CSS.

Далее ресурс подключается к системе управления. С ее помощью создаются страницы, категории и подкатегории проекта, определяется порядок работы вкладки меню, расставляются гиперссылки. Ресурс получает окончательную структуру, которая будет представлена посетителям. Сайт просто верстается на HTML и CSS языках без прикрутки к движку.

На этом же этапе верстаются категории и подкатегории, контент, который будет размещен на странице. Большое внимание верстальщик уделяет верстке карточки товара для интернет-магазина, которая должна соответствовать техническим и пользовательским требованиям.

### 1.1.5 Программирование

На этапах создания простого сайта (например, одностраничника) в программировании нет необходимости. Программист подключается к работе, если разрабатывается большой и сложный проект. Задачей специалиста стоит разработка сложных функций: калькулятора, системы оплаты с помощью банковских карт, конвертер валюты, то есть всех тех функций, которые не входят в стандартный пакет функционала.

### 1.1.6 Наполнение контентом

Предпоследний этап создания веб-сайта – наполнение страниц графическим и информационным контентом. Размещается видео, фото, тексты и другая информация, которую сможет увидеть или прочитать посетитель.

На основе семантического ядра пишутся SEO-статьи (если предполагается продвижение ресурса СЕО-методом), контент-менеджер размещает графические элементы в логической структуре. Страницы проходят внутреннюю оптимизацию.

### 1.1.7 Тестирование

Важнейшую роль в процессе создания сайта играет его тестинг. Выполняет работу тестировщик, который мониторит функциональность ресурса по ряду критериев и выявляет ошибки, которые должны быть устранены.

## 1.2 Средства разработки

Данная работа включает в себя довольно простые средства разработчика, такие как: HTML, CSS, JavaScript, Yandex.Maps API. Весь код пишется в IDE vs Code, тестируемые браузеры: Google Chrome, Opera, Microsoft Edge.

### 1.2.1 HTML

HTML (от англ. HyperText Markup Language — «язык гипертекстовой разметки») — стандартизированный язык гипертекстовой разметки документов для просмотра веб-страниц в браузере. Веб-браузеры получают HTML документ от сервера по протоколам HTTP/HTTPS или открывают с локального диска, далее интерпретируют код в интерфейс, который будет отображаться на экране монитора.

Элементы HTML являются строительными блоками HTML страниц. С помощью HTML разные конструкции, изображения и другие объекты, такие как интерактивная веб-форма, могут быть встроены в отображаемую страницу. HTML предоставляет средства для создания заголовков, абзацев, списков, ссылок, цитат и других элементов.

### 1.2.2 CSS

CSS (англ. Cascading Style Sheets «каскадные таблицы стилей») — формальный язык описания внешнего вида документа (веб-страницы), написанного с использованием языка разметки (HTML или XHTML). Также может применяться к любым XML-документам, например, к SVG или XUL.

### 1.2.3 JavaScript и API

JavaScript — мультипарадигменный язык программирования. Поддерживает объектно-ориентированный, императивный и функциональный стили. Является реализацией спецификации ECMAScript (стандарт ECMA-262).

JavaScript обычно используется как встраиваемый язык для программного доступа к объектам приложений. Наиболее широкое применение находит в браузерах как язык сценариев для придания интерактивности веб-страницам.

API ( аббр. от англ. Application Programming Interface — «программный интерфейс приложения») — описание способов (набор классов, процедур, функций, структур или констант), которыми одна компьютерная программа может взаимодействовать с другой программой. Язык JavaScript тоже является, в каком-то смысле, API.

Yandex.Maps API - это набор сервисов, которые позволяют использовать картографические данные и технологии Яндекса в ваших проектах.

# 2 ПРАКТИЧЕСКА ЧАСТЬ

## 2.1 Выбор варианта дизайна разрабатываемой информационной системы.

Дизайн клуба, можно разделить на два параметра. Первый – необходимо информативно-завлекающая информация, а во-вторых – это не должно быть слишком отталкивающе.

Так как курсовая работа, не предусматривает сложные обработки: регистрация, отправка формы, административная панель и т.д., то было принято решение, сделать лендинговый многостраничный сайт, который несет сугубо информативно-развлекающий характер.

## 2.2 Анализ структуры дизайна.

Тип верстки был выбран – блочный. Так как он подходит под различные браузеры персональных компьютеров. Так же этот вид подходит под тематику сайта. К тому же этот тип легче адаптировать под другие различные устройства, в том числе и мобильные.

2.3 Макет сайта

Так как верстка блочного типа, было решено сделать одним блоком «шапку» и меню навигации. Далее идет блок тела страницы, за ним «подвал».

Макет сайта представлен на рисунке 2.1

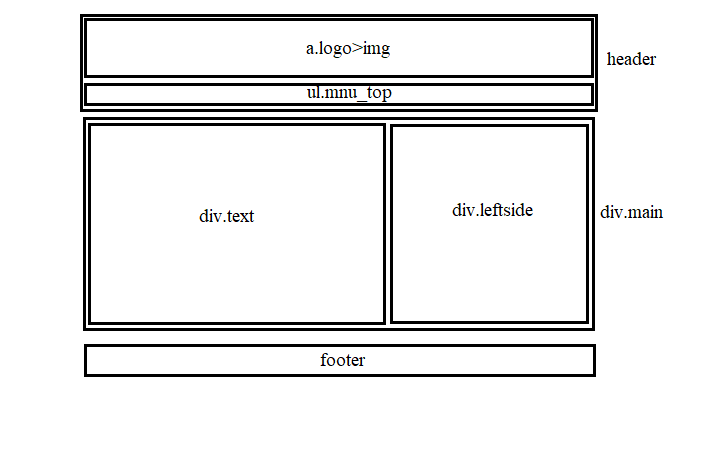


Рисунок 2.1 – Макет сайта.

В шапке страницы содержится картинка img с классом logo внутри которой, ссылка на главную страницу сайта.

Ниже располагается меню навигации по сайту, которое представлено в виде маркированного списка ul с классом mnu\_top.

Два этих блока объединены в блоке header.

Тело страницы располагается в блоке body, но было принято решение сделать отдельный блок div с классом main, для того, чтобы была возможность добавить новый элемент, который входит в body, но отделен от main.

Для более удобного расположения контента было принято решение сделать два блока div с классом text и leftside соответственно. Соотношение двух блоков 70% - text, 30% leftside.

Структура сайта в папке расположена таким образом:

Есть html файл index. html, который представляет собой главную страницу сайта.

Папка для картинок – images.

Папка со скриптами – js.

Папка для остальных HTML-старниц – html.

Папка для стилей таблиц – css.

При таком расположении файлом, можно удобно делать новые страницы с другими стилями, подключать желаемые скрипты и при этом не запутаться в одном файле.

Все же хоть и было создано несколько папок, по факту мы будем использовать 4 HTML – страницы, 1 – файл JS, 1 – файл CSS и множество картинок.

Структура сайта представлена на рисунке 2.2.

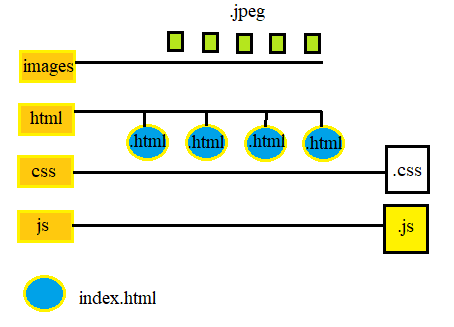


Рисунок 2.2 – Структура сайта

## 2.4 Подключение и работа с CSS

Необходимо использовать стили каскадных таблиц, постепенно. По мере необходимости, чтобы при конфликте стилей можно было восстановить рабочую версию.

Самое основное – это использовать хороший и подходящий шрифт. Подключим необходимый шрифт при помощи ресурса google fonts. Так же подключим файл .css, и будем использовать эти строки и в следующий html файлах. Листинг подключения Б.1.

Теперь необходимо сделать меню навигации, оно выполнено под тегом маркированного списка ul, где li есть элемент списка. Укажем всего 4 раздела. Листинг создания контекстного меню Б.2.

Тело страницы реализуем с помощью включения двух блоков div в один, после два блока получают свойства display: inline-block; а основной блок main – display: flex;. Размеры устанавливаются в процентах, чтобы страница могла масштабироваться под разные разрешения экранов. Реализация приведена в листинге Б.3.

Подвал, реализовать просто. В нем нет никаких ссылок, укажем ФИО студента, его почту и тему курсовой работы.

Выделим отдельный блок для указания заголовка страницы.

Итоговый вид любой html страницы будет выглядеть так, как указано на рис 2.3

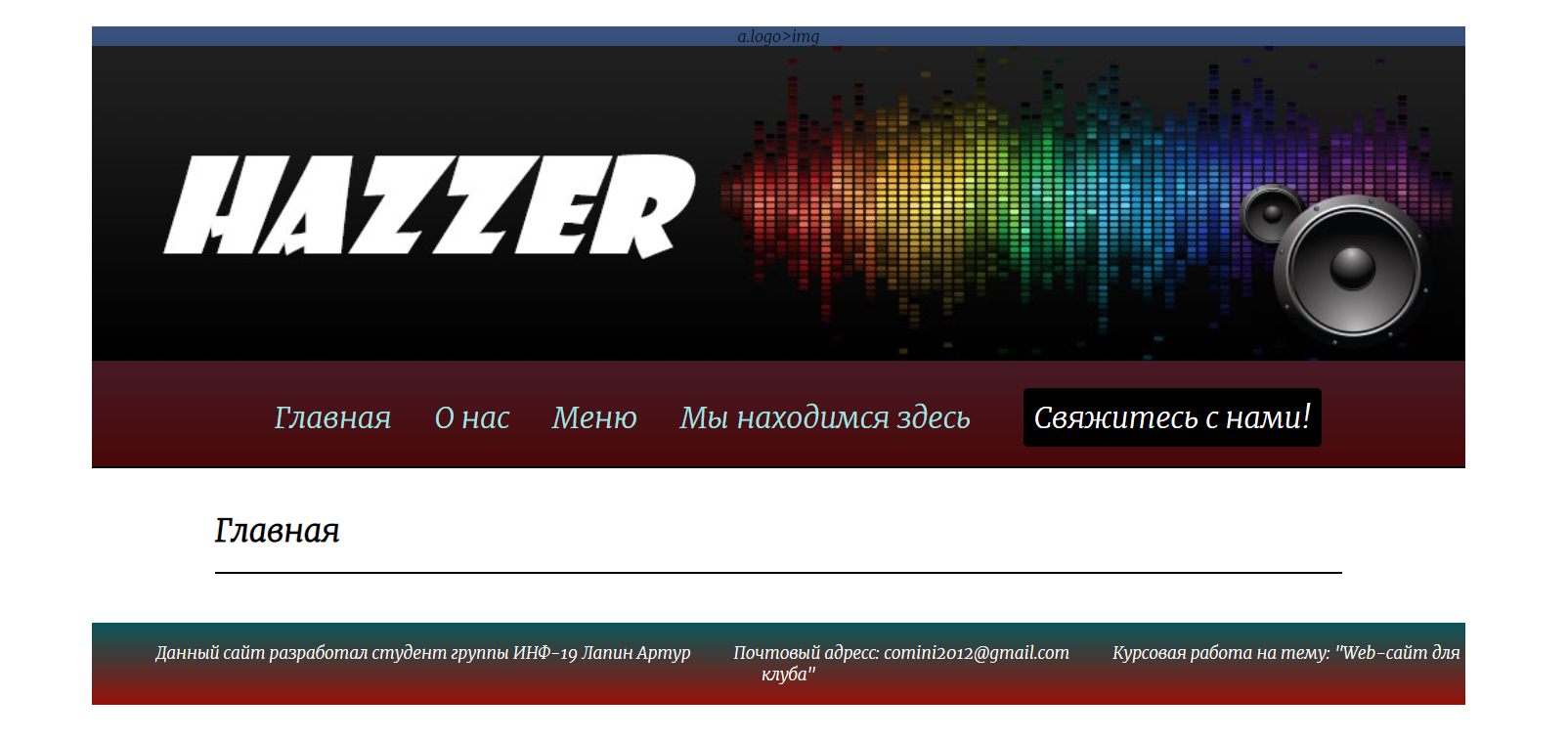


Рисунок 2.3 – Вид HTML страницы

## 2.5 Наполнение всех страниц контентом

Заполним все вкладки, определенным контентом.

На главной у нас будут справа события, слева же отведем место под таймер, который будет отсчитывать время до открытия клуба, либо же до его закрытия если он сейчас работает.

В разделе о нас, поместим текст и пару фотографий.

Меню заполним только фотографиями, где указана номенклатура и цены.

Мы находимся здесь, слева заполним информацией о заведении, справа выделим место под карту, где будет указано местоположение клуба.

«Свяжитесь с нами» - объект типа – кнопка. При нажатии на нее мы реализуем алгоритм затемнение экрана и показ наших реквизитов для связи

С заполненным контентом страницы выглядят следующим образом:

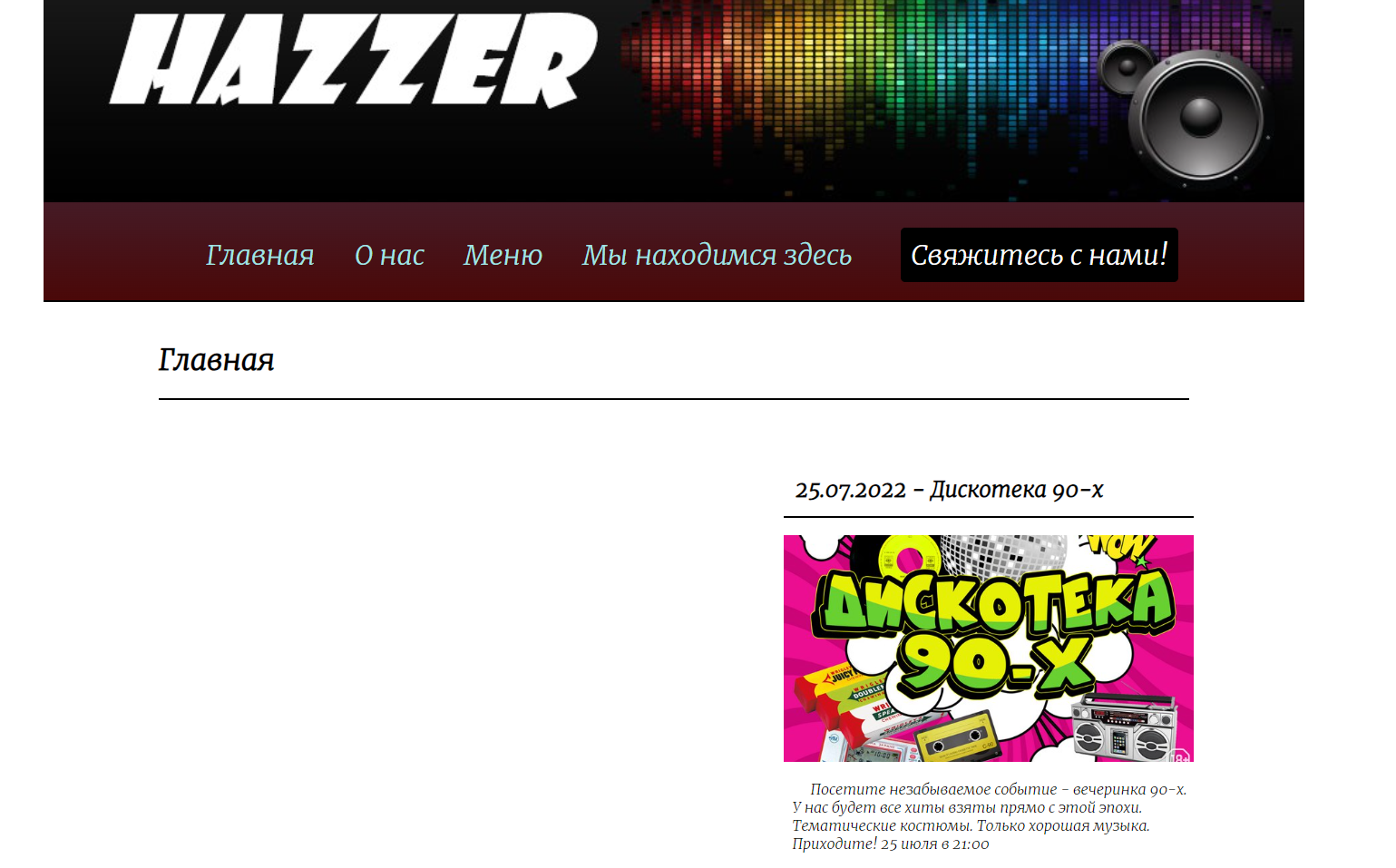
Главная – рис 2.4, о нас – рис 2.5, меню – рис 2.6, мы находимся здесь – рис 2.7.

Рисунок 2.4 – «Главная» страница с контентом

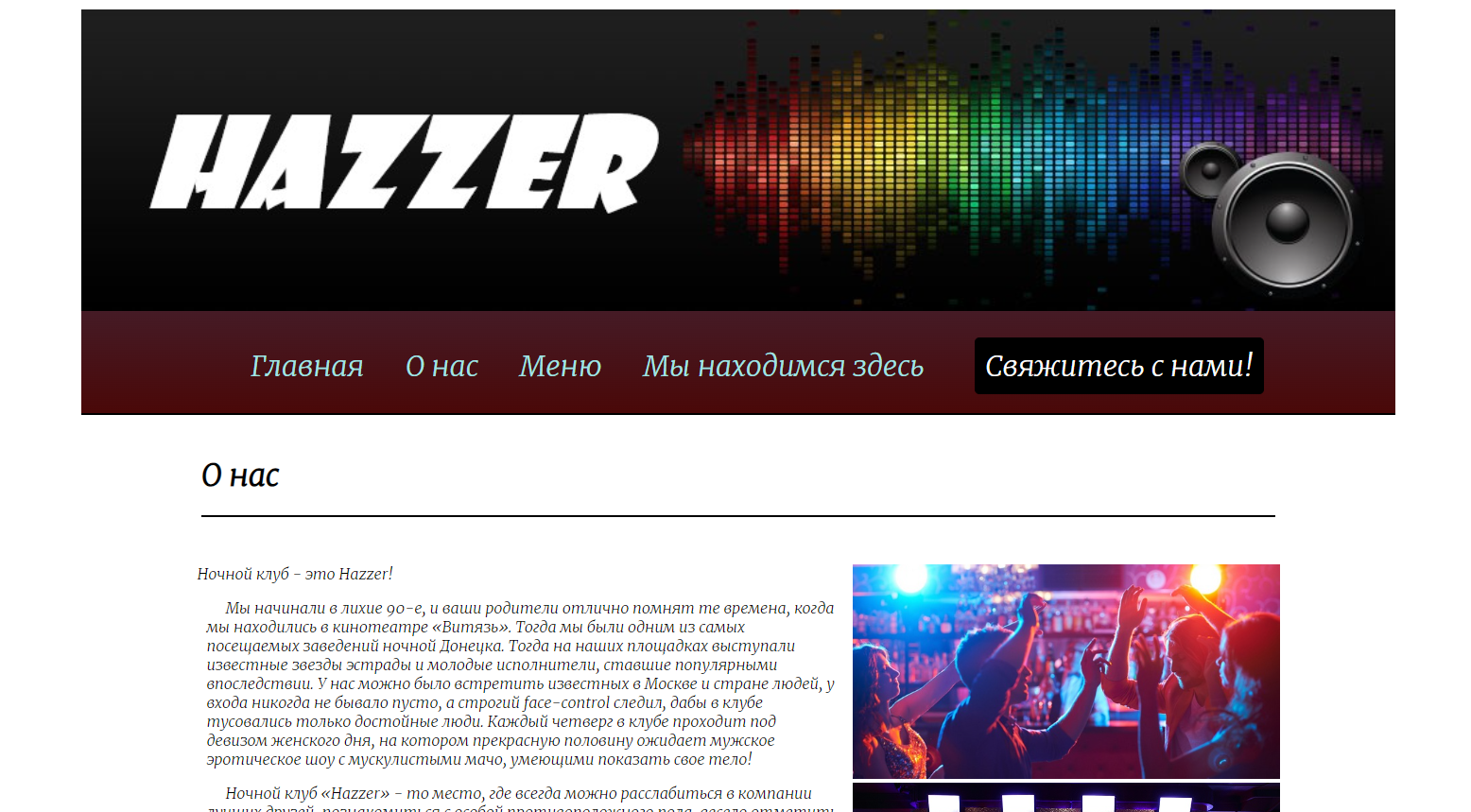


Рисунок 2.5 – «О нас» страница с контентом

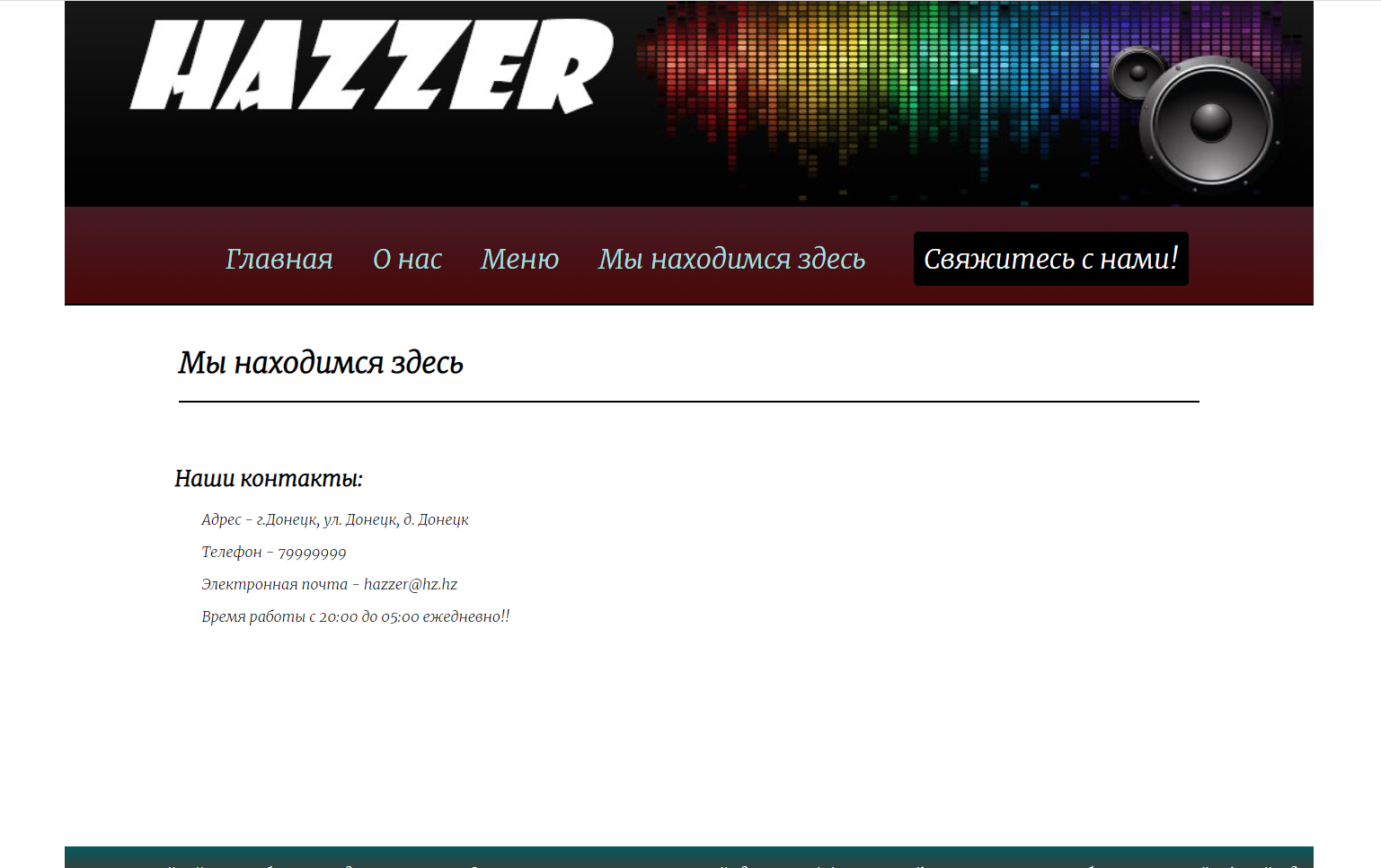


Рисунок 2.6 – «Мы находимся здесь» страница с контентом

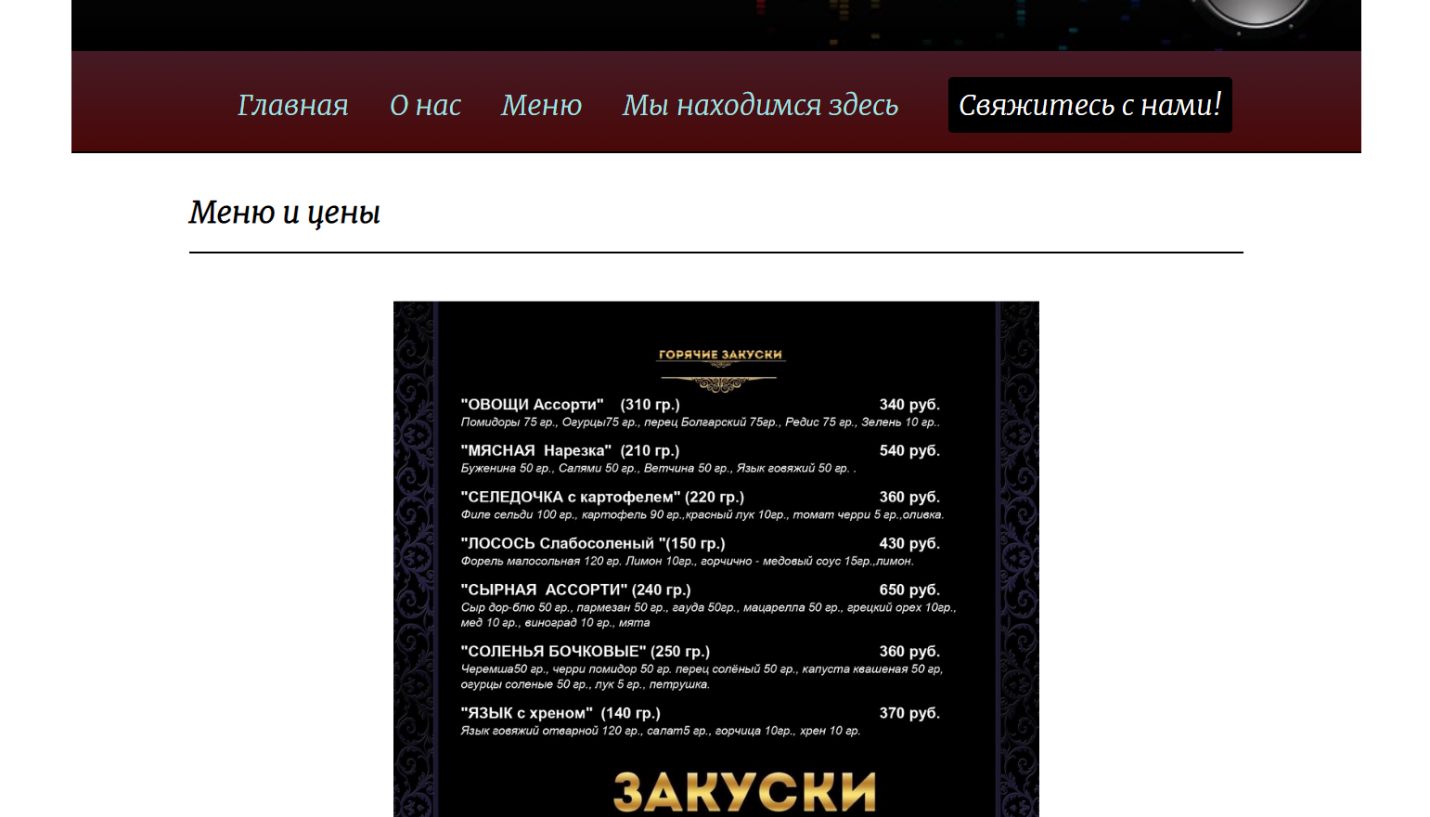


Рисунок 2.7 – «Меню» страница с контентом

## 2.6 Java Script’s

Первым делом подключим Yandex.Maps API и сделаем карту, где мы находимся на странице «Мы находимся здесь». Запишем в файл java.js функцию, которая будет работать каждый раз, при загрузки страницы и показывать координаты клуба. Листинг Б.5

При наведении на маркер клуба, всплывает окошко с его временем работы. Результат работы алгоритма приведен на рисунке 2.8

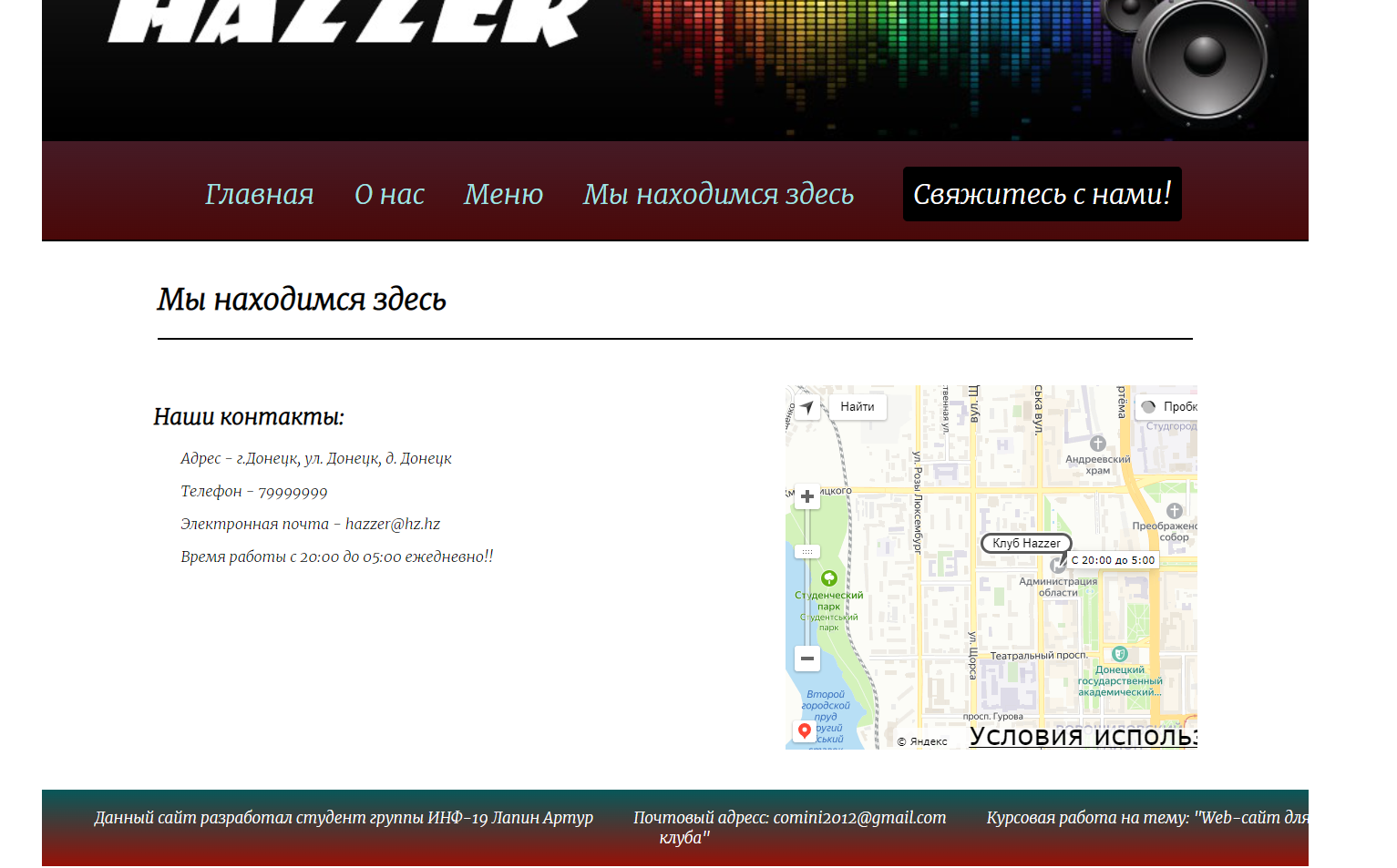


Рисунок 2.8 – «Мы находимся здесь»

Далее реализуем алгоритм кнопки «Свяжитесь с нами!». При нажатии на нее появляется информация без обновления страницы.

При помощи функции document.getElementById() выбираем id элемента, после чего просто другой функцией – classList.toggle() переключаем класс, у которого есть атрибут visible. Мы получаем, что при нажатии на кнопку мы делаем элемент видимым, добавим другую кнопку ‘ok’, которая делает абсолютно тоже самое.

При нажатии на кнопку появляется элемент, который отредактирован с помощью CSS. Результат на рис 2.9.

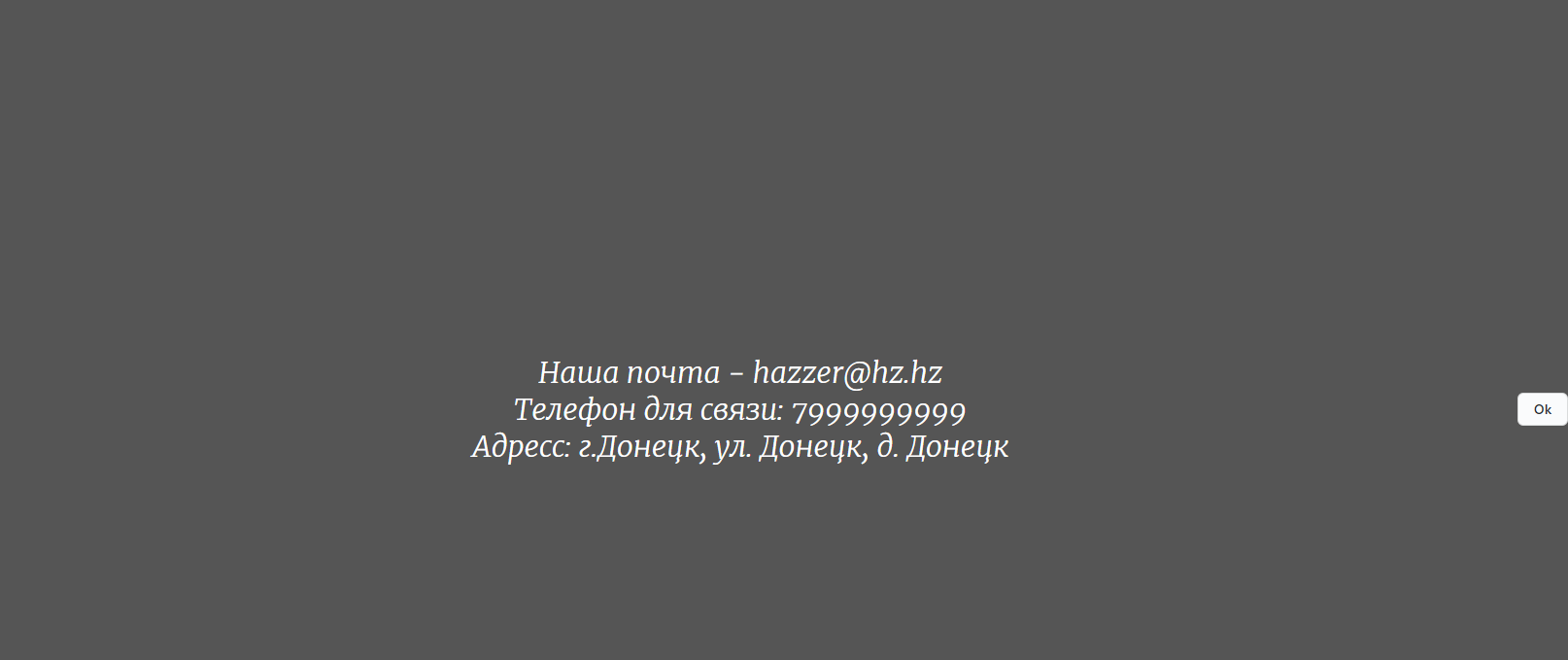


Рисунок 2.9 – Появление элемента на кнопку «Свяжитесь с нами!»

Добавим таймер на главную страницу, который будет отсчитывать время, до открытия клуба. Если клуб открыт, то он будет показывать сколько осталось до закрытия.

При помощи функции Date(), манипулируем временем. А потом при помощи функций:

* Object.classList.contains() – проверяем есть ли класс, у объекта;
* Object.classList.remove() – удаляем класс объекту;
* Object.classList.add() – добавляем класс объекту.

Мы пишем сценарий, при каких условиях показывать тот или иной элемент. Листинг Б.7

Результат работы на рисунке 2.10 и 2.11

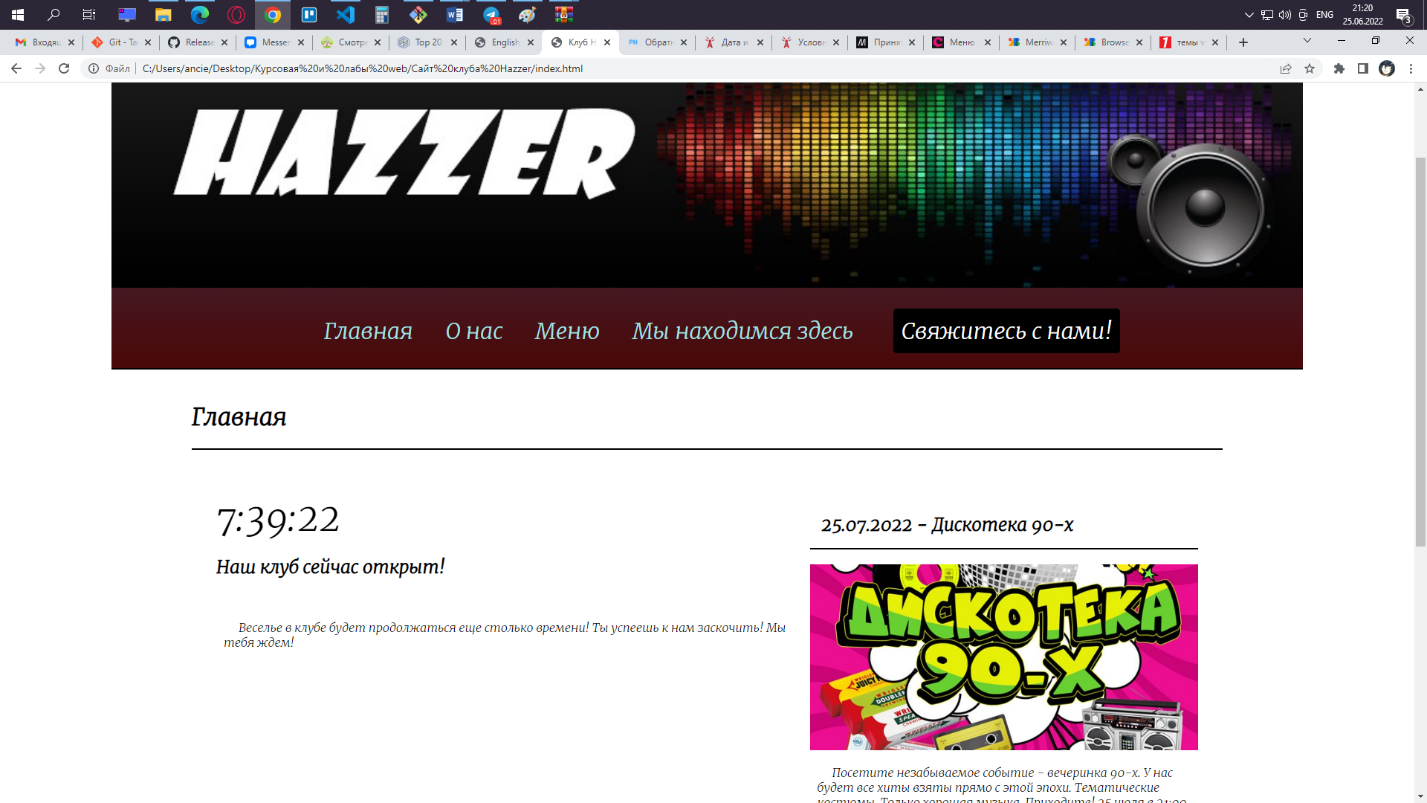


Рисунок 2.10 – Таймер, когда закроется клуб

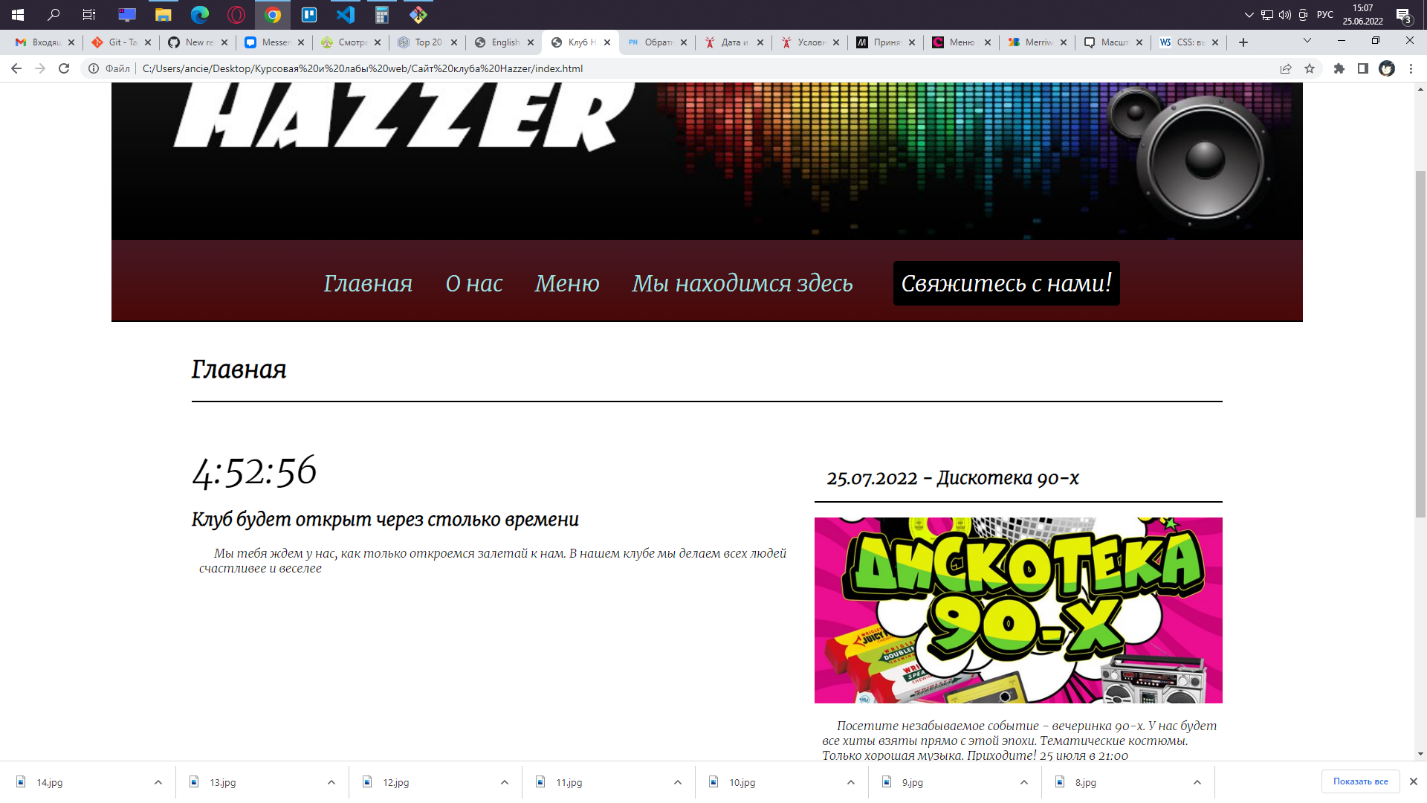


Рисунок 2.11 – Таймер, когда откроется клуб

# ВЫВОДЫ

Блочная верстка самая оптимальная для таких типов сайтов, не лучше адаптивной, но результат вышел оптимальным.

Так же при правильном использовании HTML, CSS, JavaScript можно вполне неплохо сделать лендинг.

На сегодняшний день, разработка сайтов и веб приложений, является очень востребованным навыком. При отсутствии больших временных затрат можно разработать сайт, который является необходимость на текущий момент, для любого предприятия, бизнеса, учреждения. Это лицо, а также возможность расширить своё влияние и популярность.

Поставленные требования были выполнены, соответственно можно сделать вывод, что поставленная цель выполнена

# ПЕРЕЧЕНЬ ИСПОЛЬЗОВАНЫХ ИСТОЧНИКОВ

1. Таненбаум Э. Современные операционные системы / Э.Таненбаум 3-е изд. –СПб.: Питер, 2010. – 1120 с.: ил.

2. Харт, Джонсон, М. Системное программное обеспечение в среде Windows / Харт, Джонсон, М., 3-е издание. :Пер.с англ. М. Издательский дом «Вильямс», 2005. – 592 с.:ил.

3. Олифер Н. А. Компьютерные сети. Принципы, технологии, протоколы: Учебник для вузов / Н. А.Олифер, В. Г. Олифер 4-е изд. - СПб.: Питер, 2011. – 944 с.: ил.

4. Олифер Н. А., Олифер В. Г. Сетевые операционные системы/ Н.А.Олифер, В. Г. Олифер - СПб.: Питер, 2007. – 864 с.: ил.

5. .Мешков А.,Тихонов Ю. Visual C++ и MFC. Программирование для Windows NT и Windows 95-:В трех томах. Том 1. / А.Мешков, Ю.Тихонов –СПб.:BHV- Санкт-Петербург,1997.-464с.,ил.

6. Кузминьский М., Волков Д. Современные суперкомпьютеры: состояние и перспективы / М. Кузминьский, Д. Волков – СПб: Питер, 2005. – 324 с., ил.

7. Воеводин В. В. Суперкомпьютерная грань компьютерного мира / В. В. Воеводин – СПб: Питер,2009 – 188 с.

8. Деревянко А.С.Операционные системы: Учебное пособие. / А.С.Деревянко , М.Н. Солощук– Х.: НТУ ХПИ, 2002. - 573c.

# Приложение Б ЛИСТИНГИ

## Б.1 – Подключение google fonts и .css файла. .html, style.css

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

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

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

       <link href="https://fonts.googleapis.com/css2?family=League+Gothic&family

=Merriweather:ital,wght@1,300&display=swap" rel="stylesheet">

.html

body{

    font-family: 'League Gothic', sans-serif;

    font-family: 'Merriweather', serif;

}

## Б.2 – Реализация горизонтального меню на сайте style.css

ul{

    width: 100%;

    list-style: none;

    margin: 0;

    padding: 20px;

}

li{

    display: inline;

}

## Б.3 – Реализация тела страницы style.css

.leftside{

    overflow: hidden;

    height: 100%;

    display:inline-block;

    width: 40%;

    margin-left: 15px;

}

.main{

    display:flex;

        max-width: 100%;

        height: auto

}

.text{

    display:inline-block;

    width: 60%;

}

## Б.4 – Реализация подвала style.css

footer {

    max-width: 85%;

    margin: 44px auto;

    font-size: 17px;

    background: linear-gradient(hsl(181, 85%, 19%),hsl(4, 92%, 31%))

}

## Б.5 – Листинг java.js

ymaps.ready(init);

function init() {

    var myMap = new ymaps.Map("map", {

            center: [48.00878157, 37.79983749],

            zoom: 15

        }, {

            searchControlProvider: 'yandex#search'

        }),

        myGeoObject = new ymaps.GeoObject({

            geometry: {

                type: "Point",

                coordinates: [48.00878157, 37.79983749]

            },

            properties: {

                iconContent: 'Клуб Hazzer',

                hintContent: 'С 20:00 до 5:00'

            }

        }, {

            preset: 'islands#blackStretchyIcon',

            draggable: false

        })

    myMap.geoObjects.add(myGeoObject);

}

## Б.6 – Листинг кнопки «Свяжитесь с нами»

<script>

        function myFunction() {

            var popup = document.getElementById("myPopup");

            popup.classList.toggle("show");

          }

        </script>

.show {

    visibility: visible;

    -webkit-animation: fadeIn 1s;

    animation: fadeIn 1s;

  }

## Б.7 – Листинг таймера

<script>

            let timer\_show = document.getElementById("timer");

            let timerop\_show = document.getElementById("timerop");

            let No = document.getElementById("hideNO");

            let At = document.getElementById("hideAT");

function diffSubtract(date1, date2) {

    return date2 - date1;

}

let end\_date = {

    "hours": "2",

    "minutes": "00",

    "seconds": "00"

}

let start\_date = {

    "hours": "17",

    "minutes": "00",

    "seconds": "00"

}

let nowday = new Date();

let sdf = nowday.getHours();

if (sdf < 5) {

    let end\_date\_str = `${nowday.getFullYear()}.${nowday.getDay()}.${nowday.getDate()}.${end\_date.hours}:${end\_date.minutes}:${end\_date.seconds}`;

    let start\_date\_str = `${nowday.getFullYear()}.${nowday.getDay()}.${nowday.getDate()}.${start\_date.hours}:${start\_date.minutes}:${start\_date.seconds}`;

timer = setInterval(function () {

    let now = new Date();

    let nnow = `${now.getFullYear()}.${now.getDay()}.${now.getDate()}.${now.getHours()}:${now.getMinutes()}:${now.getSeconds()}`;

    let ndate = new Date(nnow);

    let date = new Date(end\_date\_str);

    let dateopen = new Date(start\_date\_str);

    let ms\_left = diffSubtract(ndate, date);

    let ms\_leftop = diffSubtract(ndate, dateopen);

    let res = new Date(ms\_left);

    let resop = new Date(ms\_leftop);

    let str\_timer = `${res.getHours()}:${res.getMinutes()}:${res.getSeconds()}`;

    let str\_timerop = `${resop.getHours()}:${resop.getMinutes()}:${resop.getSeconds()}`;

    timer\_show.innerHTML = str\_timer;

    timerop\_show.innerHTML = str\_timerop;

}, 1000)}

if (sdf > 5) {

    let end\_date\_str = `${nowday.getFullYear()}.${nowday.getDay()}.${nowday.getDate() +1}.${end\_date.hours}:${end\_date.minutes}:${end\_date.seconds}`;

    let start\_date\_str = `${nowday.getFullYear()}.${nowday.getDay()}.${nowday.getDate()}.${start\_date.hours}:${start\_date.minutes}:${start\_date.seconds}`;

timer = setInterval(function () {

    let now = new Date();

    let nnow = `${now.getFullYear()}.${now.getDay()}.${now.getDate()}.${now.getHours()}:${now.getMinutes()}:${now.getSeconds()}`;

    let ndate = new Date(nnow);

    let date = new Date(end\_date\_str);

    let dateopen = new Date(start\_date\_str);

    let ms\_left = diffSubtract(ndate, date);

    let ms\_leftop = diffSubtract(ndate, dateopen);

    let res = new Date(ms\_left);

    let resop = new Date(ms\_leftop);

    let str\_timer = `${res.getHours()}:${res.getMinutes()}:${res.getSeconds()}`;

    let str\_timerop = `${resop.getHours()}:${resop.getMinutes()}:${resop.getSeconds()}`;

    timer\_show.innerHTML = str\_timer;

    timerop\_show.innerHTML = str\_timerop;

    if (ndate.getHours() > 20 || ndate.getHours() < 5 ){

        var oR = true;

    } else {

        var oR = false;

    }

    if (oR == true){

        if (timer\_show.classList.contains("openNow") == true){

        timer\_show.classList.remove("openNow");

        timer\_show.classList.add("opensn");

        }

        if (At.classList.contains("aOpen") == true){

            At.classList.remove("aOpen");

            At.classList.add("AtOpen");

        }

        if (No.classList.contains("NowOpen")){

            No.classList.remove("NowOpen");

            No.classList.add("nOpen");

        }

        if (timerop\_show.classList.contains("opensa") == true){

        timerop\_show.classList.remove("opensa");

        timerop\_show.classList.add("openAt");

        }

    }

    if (oR == false){

        if (timerop\_show.classList.contains("openAt") == true){

        timerop\_show.classList.remove("openAt");

        timerop\_show.classList.add("opensa");

        }

        if (At.classList.contains("AtOpen") == true){

            At.classList.remove("AtOpen");

            At.classList.add("aOpen");

        }

        if (No.classList.contains("nOpen")){

            No.classList.remove("nOpen");

            No.classList.add("NowOpe");

        }

        if (timer\_show.classList.contains("opensn") == true){

        timer\_show.classList.remove("opensn");

        timer\_show.classList.add("openNow");

        }

    }

}, 1000)}

        </script>