**Частное образовательное учреждение высшего образования**

**«Международный Институт Дизайна и Сервиса»**

**(ЧОУВО МИДиС)**

Кафедра математики и информатики

**КУРСОВАЯ РАБОТА**

по дисциплине: «Разработка приложений для Microsoft NET Framework на тему: «**Разработка веб-сайта на платформе ASP.NET»**

Направление подготовки 09.03.03 Прикладная информатика,

направленность (профиль) Прикладная информатика в экономике

Научный руководитель

доцент КМиМ

С.С.Чеботарев

Автор работы

Студент группы ПИ-324

Кочкина Анастасия Алексеевна

Челябинск 2022

**Содержание**

Оглавление

[Введение 3](#_Toc149062210)

[ГЛАВА 1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 5](#_Toc149062211)

[1.1 Описание предметной области 5](#_Toc149062212)

[1.2 Выделение словаря предметной области 6](#_Toc149062213)

[1.3 Виды связей между объектами 7](#_Toc149062214)

[1.3.1 Связь «один к одному» 7](#_Toc149062215)

[1.3.2 Связь «один ко многим» 7](#_Toc149062216)

[1.3.3 Связь «многие ко многим» 7](#_Toc149062217)

[1.4 Анализ существующих программных решений 8](#_Toc149062218)

[1.5 Техническое задание 16](#_Toc149062219)

[1.5.1 Общие сведения 16](#_Toc149062220)

[1.5.2 Назначение системы 17](#_Toc149062221)

[1.5.3 Характеристика объекта автоматизации 17](#_Toc149062222)

[1.5.4 Требования к системе 18](#_Toc149062223)

[1.5.5 Состав и содержание работ по созданию системы 19](#_Toc149062224)

[1.5.6 Условия эксплуатации 19](#_Toc149062225)

[2.1 Проектирование и разработка классов 19](#_Toc149062226)

[2.2.1 Главная страница 24](#_Toc149062227)

[2.2. 2 Страница со списком услуг 26](#_Toc149062228)

[2.2. 3 Страница со списком специалистов 26](#_Toc149062229)

[2.2.4 Страница отзывов 28](#_Toc149062230)

[2.2.5 Местонахождение клиники 29](#_Toc149062231)

[2.2.6 Корзина пациента 30](#_Toc149062232)

[2.2.7 Главная страница администратора 31](#_Toc149062233)

[2.2.8 Нанять врача 32](#_Toc149062234)

[2.2.9 Редактирование категорий 32](#_Toc149062235)

[2.2.10 Роли 33](#_Toc149062236)

[2.3 Описание и разработка базы данных 34](#_Toc149062237)

[2.4 Краткое руководство пользователя по эксплуатации системы 36](#_Toc149062238)

[ЗАКЛЮЧЕНИЕ 36](#_Toc149062239)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 37](#_Toc149062240)

# Введение

«Разработка веб-сайта на платформе ASP.NET - так звучит название данной курсовой работы. В данной работе спроектирована разработка сайта по отрасли частной стоматологии. В современном мире происходят такая тенденция, как постоянно и непрерывно увеличивающийся поток информации. В связи с этим появляется острая необходимость в процессах, которые могли бы приводить информацию к общепринятому виду, а также в автоматизации рутинной, монотонной работы. Проектирование и разработка веб-сайта позволит увеличить производительность и эффективность труда, а также качество и скорость работы. Именно поэтому эта тема является актуальной. Этот проект может обеспечивать хранение данных, оптимизировать процессы, связанные с пациентами и сотрудниками. И обеспечивать хранение и использование разных ролей в системе, таких как администратор, обычный пользователь и многое другое.

Цель работы – разработка веб-сайта с использованием MVC на основе ASP.NET и различных инструментов для области клиента, при помощи объектно-ориентированного подхода к программированию на C#.

Были поставлены следующие задачи, для достижения основной цели:

* провести системный анализ предметной области;
* составить техническое задание для решения задачи;
* спроектировать и разработать классы для хранения и представления данных и моделей;
* разработать пользовательский интерфейс согласно дизайн-макетам;
* выполнить реализацию приложения с использованием современных средств программирования, а также его тестирование и отладку.

Объектом исследования данной курсовой работы являются методы и технологии разработки программных продуктов с применения объектно-ориентированного программирования и языка программирования С#.

Предметом исследования данной курсовой работы являются методы, алгоритмы и приёмы разработки веб-сайта на основе ASP.NET «Стоматология».

Информационной базой исследования является учебная литература по информатике и программированию, техническая документация по языку С# инструментальной среды MS Visual Studio 2022, а также использование технологии Blazor.

# ГЛАВА 1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

# Описание предметной области

Предметная область в разных сферах обозначает разное, начиная с «научной теории», и, заканчивая «логикой». В информационной среде предметная область – часть реального мира, которая подлежит изучению с целью автоматизации организации управления. В данной работе название предметной области – «Стоматология». Объектами являются лица, события или предметы, о которых ниже представлена информация.

В данной работе предметную область составляют:

Специалисты характеризуются такими параметрами, как:

* Фамилия Имя Отчество
* Описание
* Категория, в которой они специализируются
* Фото

Список услуг характеризуется следующими параметрами:

* Название услуги
* Цена

Под субъектами предметной области данного проекта будем понимать:

Пользователя, который может использовать следующие операции:

* Просмотреть более детальную информации каждой услуги;
* Просмотреть более детальную информацию по каждому выбранному врачу;
* Записаться на выбранную услугу и получить ее отображение у себя в корзине;
* Записаться к выбранному врачу и получить его отображение у себя в корзине;
* Зарегистрировать свой аккаунт или войти в существующий;
* Оставить отзыв о клинике и специалистах;
* Написать в чате менеджеру и задать интересующие вопросы

Администратора, который может пользоваться следующими функциями:

* Нанять нового врача;
* Уволить врача;
* Редактировать список категорий, к которым относятся услуги и врачи
* Изменять роли и права доступа

Контент-менеджера, у которого есть следующие функции:

* ???????????????????????????????????????????????

# 1.2 Выделение словаря предметной области

Пациент – это пользователь, который пользуется услугами на сайте и записывается на приём.

Специалист – человек, который имеет врачебное образование и лечит зубы.

Услуга – предназначена для лечения той или иной области зубного характера.

Администратор – человек, имеющий свой отдельный аккаунт на сайте и имеющий больше возможностей, чем пользователь для более организованной работы системы.

Контент-менеджер – человек, имеющий свой отдельный аккаунт и отвечающий за содержимое на сайте.

Отзыв – субъективное мнение пациентов по окончании работы специалистом.

Корзина – место хранения талонов к записанным врачам и демонстрация услуг, на которые записан пользователь.

Категория – сфера деятельности каждого специалиста, к которой он относится.

# 1.3 Виды связей между объектами

Всего существует три вида взаимосвязей между объектами.

•связь «один к одному»;  
 • связь «один ко многим»;

• связь «многие ко многим».

# 1.3.1 Связь **«**один к одному**»**

**Связь «один к одному»** – это ситуация, при которой одному объекту таблицы А соответствует один объект таблицы Б, и наоборот.

# 1.3.2 Связь **«**один ко многим**»**

**Связь «один ко многим»** - это ситуация, при которой объекту А может принадлежать или же соответствовать несколько объектов Б, но объекту Б может соответствовать только один объект А.

В соответствии с рис.2, одному врачу могут соответствовать несколько категорий.



Рисунок 1 - Связь "один ко многим"

# 1.3.3 Связь **«**многие ко многим**»**

**Связь «многие ко многим» -** это ситуация, при которой нескольким объектам из таблицы А может соответствовать несколько объектов из таблицы Б, и в тоже время нескольким объектам из таблицы Б соответствует несколько объектов из таблицы А.

# 1.4 Анализ существующих программных решений

Для примера похожий программных решений я взяла такие проекты, как «Стоматология на Вернадского» с использованием фреймворка ASP.NET и «Мастер-Дент» (рис. 6).

Программное обеспечение «Стоматология на Вернадского» (рис.2) с продуманной архитектурой и интуитивно понятной навигацией.

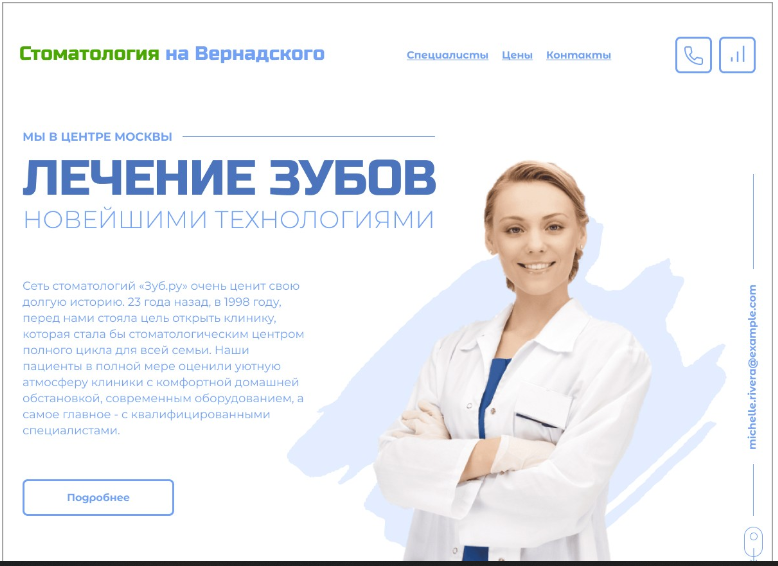


Рисунок 2 - Стоматология на Вернадского

На сайте присутствуют такие модульные элементы, как:

* каталоги различных уровней вложения;
* видео- и фотогалереи;
* интерактивная карта для удобного поиска филиалов клиники;
* форма записи на приём;
* всплывающие окна с информацией об услугах и сотрудниках.

Для сайта разработаны мобильные версии для корректного отображения на любом мобильном устройстве.

Сайт хорошо оптимизирован и хорошо взаимодействует с поисковыми системами.

Был проделан анализ возможной структуры сайта, чтобы потенциальный клиент мог легко и удобно передвигаться по сайту и, в конечном счете, получить всю необходимую информацию об услугах компании, выбрать подходящего специалиста и записаться на приём.

Были сделаны прототипы основных страниц:

* Страница каталога услуг с описаниями услуг и ценами (рис.3)



Рисунок 3 - услуги с ценами

* Каталог специалистов с детальным описанием каждого (рис.4);

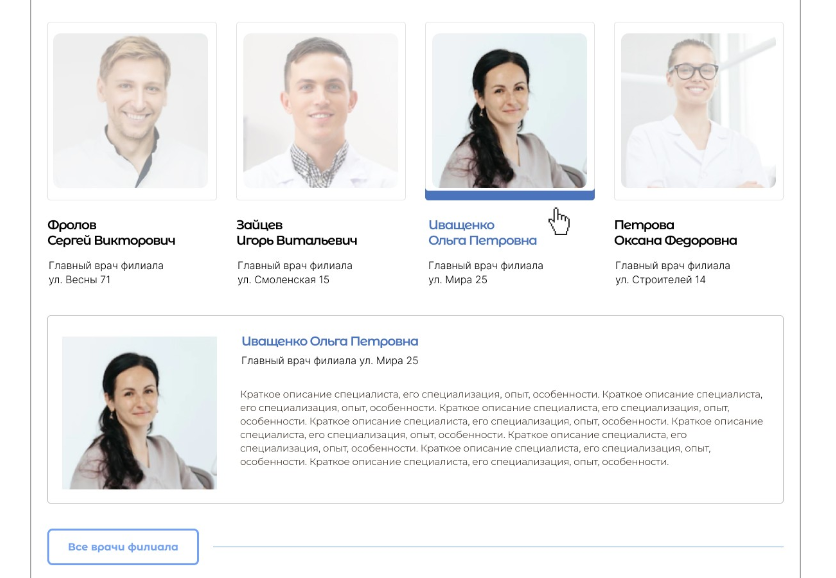


Рисунок 4 – специалисты

* Форма записи на прием и контактная информация (рис.5)

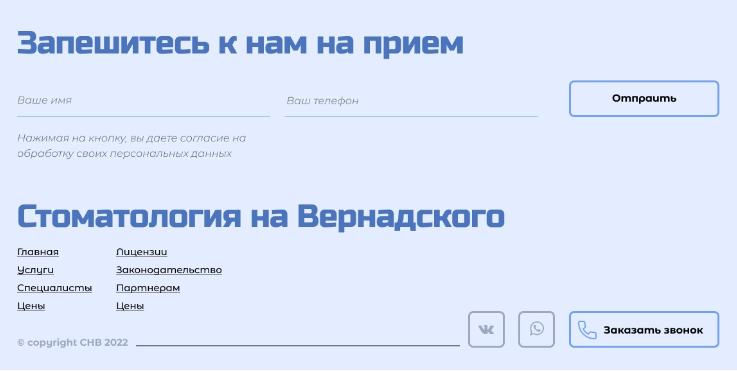


Рисунок 5 - форма записи

Для сайта разработан индивидуальный, яркий и запоминающийся дизайн, позволяющий выгодно презентовать компанию на фоне конкурентов и представить информацию о своих услугах.

Перейдем к рассмотрению второго программного обеспечения – «Мастер-Дент». (рис.6). Этот веб-сайт предлагает возможность получения стоматологических услуг.

Программный продукт «Мастер-Дент» автоматизирует и предлагает группы процессов, которые охватывают всю деятельность стоматологии, а именно:

* Оформление слайдера акций для удобства клиентов (рис. 7)

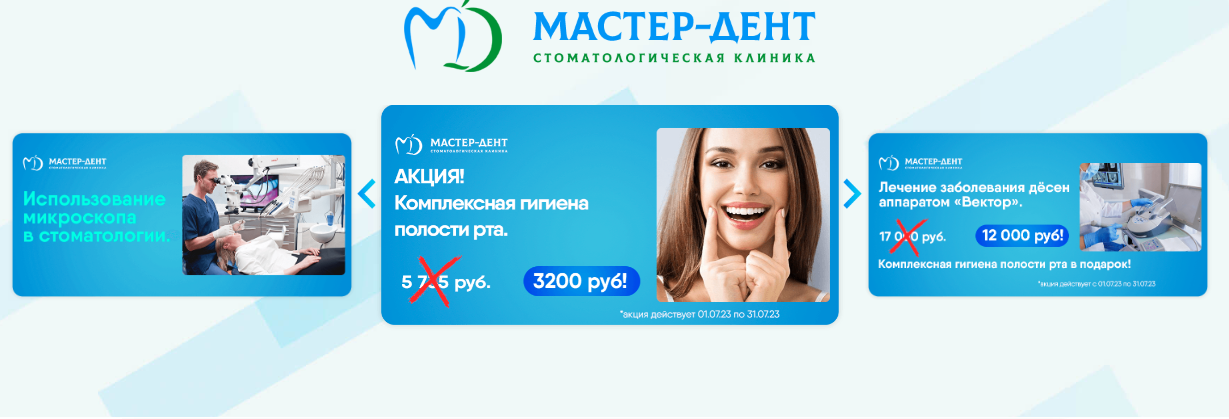


Рисунок 6 - слайдер

* Удобные и понятные вкладки по виду услуг, на которые можно записаться (рис. 7)

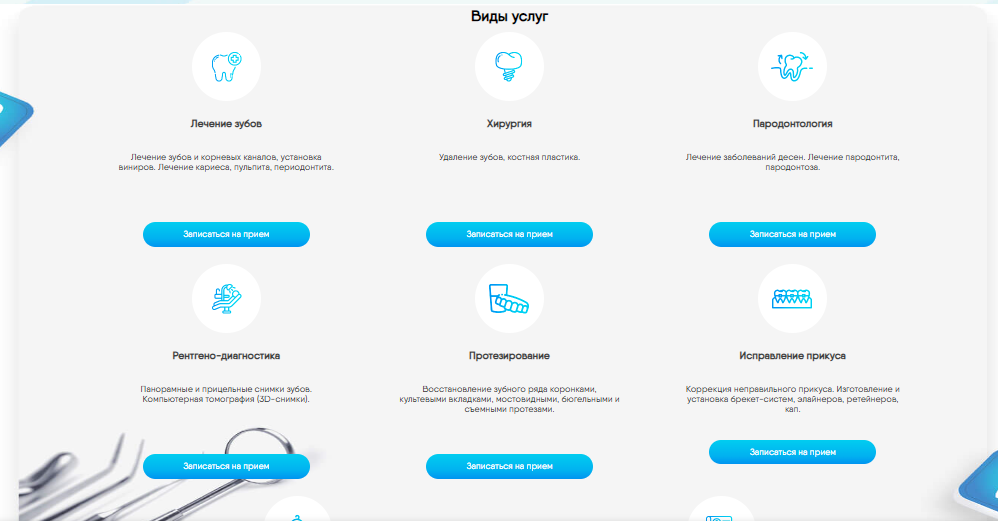


Рисунок 7 - виды услуг

* Автоматизация функции записи на прием с помощью заполнения доступной формы (рис. 8);

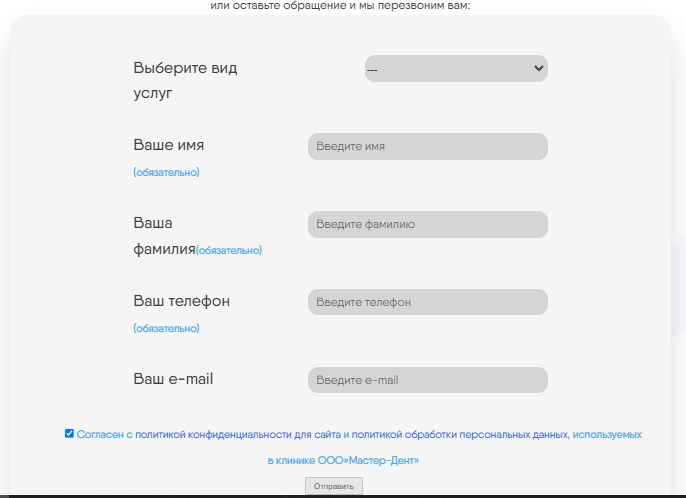


Рисунок 8 - форма

* **Также на сайте расположена удобная карта с местоположением клиники, что упрощает многие вопросы пациентов (рис. 9)**

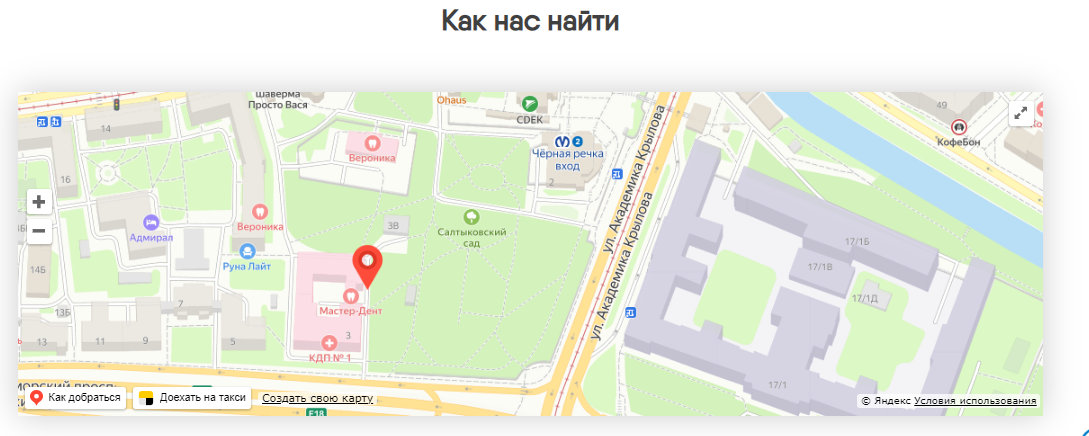


Рисунок 9 - местоположение

Программные средства, используемые для решения задач:

**Microsoft Visual Studio –** этоинтегрированная среда разработки программного обеспечения от компании Microsoft, предназначенная разрабатывать как консольные приложения, так и игры, и приложения с графическим интерфейсом, в том числе с поддержкой технологии Windows Forms, UWP, а также веб-сайты, веб-приложения и веб-службы.

**SQL Server Management Studio (SSMS)** — это интегрированная среда для управления любой инфраструктурой SQL. Используйте SSMS для доступа, настройки, администрирования и разработки всех компонентов SQL Server.

Объектно-ориентированный язык С# создан специально для работы с фреймворком .NET (Windows Forms, WPF, ASP.NET, Xamarin).

Также были использованы инструменты – Blazor для разработки клиентской части и библиотека SignalR.

C# относится к семье языков с С-подобным синтаксисом, из них его синтаксис наиболее близок к [C++](https://ru.wikipedia.org/wiki/C%2B%2B) и Java. Язык имеет статическую типизацию, поддерживает полиморфизм, перегрузку операторов (в том числе операторов явного и неявного приведения типа), делегаты, атрибуты, [события](https://ru.wikipedia.org/wiki/%D0%A1%D0%BE%D0%B1%D1%8B%D1%82%D0%B8%D0%B9%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5), [переменные](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D0%B5%D0%BC%D0%B5%D0%BD%D0%BD%D0%B0%D1%8F_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), [свойства](https://ru.wikipedia.org/wiki/%D0%A1%D0%B2%D0%BE%D0%B9%D1%81%D1%82%D0%B2%D0%BE_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), [обобщённые](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D0%BE%D0%B1%D1%89%D1%91%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) типы и методы, [итераторы](https://ru.wikipedia.org/wiki/%D0%98%D1%82%D0%B5%D1%80%D0%B0%D1%82%D0%BE%D1%80), [анонимные функции](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%BE%D0%BD%D0%B8%D0%BC%D0%BD%D0%B0%D1%8F_%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F) с поддержкой [замыканий](https://ru.wikipedia.org/wiki/%D0%97%D0%B0%D0%BC%D1%8B%D0%BA%D0%B0%D0%BD%D0%B8%D0%B5_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), [LINQ](https://ru.wikipedia.org/wiki/Language_Integrated_Query), [исключения](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B0_%D0%B8%D1%81%D0%BA%D0%BB%D1%8E%D1%87%D0%B5%D0%BD%D0%B8%D0%B9), [комментарии](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BC%D0%B5%D0%BD%D1%82%D0%B0%D1%80%D0%B8%D0%B8_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) в формате [XML](https://ru.wikipedia.org/wiki/XML).

Переняв многое от своих предшественников языков [C++](https://ru.wikipedia.org/wiki/C%2B%2B), [Delphi](https://ru.wikipedia.org/wiki/Delphi_(%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F)), [Модула](https://ru.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D1%83%D0%BB%D0%B0-2), [Smalltalk](https://ru.wikipedia.org/wiki/Smalltalk" \o "Smalltalk) и, в особенности, [Java](https://ru.wikipedia.org/wiki/Java) — С#, опираясь на практику их использования, исключает некоторые модели, зарекомендовавшие себя как проблематичные при разработке программных систем, например, C# в отличие от C++ не поддерживает множественное наследование классов (между тем допускается [множественная реализация интерфейсов](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_(%D0%BE%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)#%D0%9C%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2%D0%B5%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BD%D0%B0%D1%81%D0%BB%D0%B5%D0%B4%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_%D0%B8_%D1%80%D0%B5%D0%B0%D0%BB%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F_%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%8)). C# изначально был создан языком со сложной архитектурой для решения сложных задач, и программирования больших программных обеспечений.

Объектно-ориентированные языки программирования отличаются от функциональных или процедурных языков. Функциональное или процедурное программирование идеально работает в простых программах. Функции аккуратно вложены друг в друга, взаимодействуют друг с другом, можно передать данные из одной функции в другую. Особенность таких [языков программирования](https://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) состоит в том, что задачи разбиваются на шаги и решаются шаг за шагом.

Используя процедурный язык, программист определяет языковые конструкции для выполнения последовательности алгоритмических шагов. К процедурным языкам программирования относятся языки 1С, Фортран, Си и т.д. В этих языках используется оператор GOTO. В его основе – процедуры и функции. Объектно-ориентированные языки основаны на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определённого класса. К объектно-ориентированным языкам кроме С# относятся такие языки, как PHP. PYTHON, JAVA, JAVA SCRIPT и т.д.

Основные принципы структурирования в случае ООП связаны с различными аспектами базового понимания предметной задачи, которое требуется для оптимального управления соответствующей моделью:

* **Инкапсуляция** позволяет открыть только ту часть функций и данных, которая нужна для внешних пользователей, а остальное спрятать внутри класса.
* Модификатор доступ**а** – ключевые слова в объектно-ориентированных языках, которые задают параметры доступа для классов, методов и прочих элементов. Они определяют степень доступности объектов класса извне. В языке С# реализуются следующие модификаторы доступа: public. private и т.д.
* **Наследование** позволяет делать производные структуры на основе базовых, тем самым давая возможность осуществлять повторное использование этих структур.
* «Абстрактный класс» – это родительский класс, который содержит общие абстрактные атрибуты. На его основе нельзя создать объект, но его может унаследовать родительский класс.
* Полиморфизм – принцип ООП, когда один фрагмент кода может работать с разными типами в C# реализуется с помощью перегрузок. П**олиморфизм**позволяет программировать на основе интерфейсов, у которых может быть несколько реализаций. Полиморфизм осуществляется в объектно-ориентированных-языках путём использования виртуальных методов, что является очень удобным и безопасным. Полиморфизм – это **ключевое свойство ООП** для построения архитектуры. Он позволяет сделать модуль независимым от конкретной реализации интерфейса. Этот принцип называется **инверсией зависимостей**, на котором основаны все плагинные системы.
* Инверсия зависимостей так называется, что она позволяет изменить направление зависимостей. Сначала начинаем писать в простом стиле, когда высокоуровневые функции зависят от низкоуровневых. Однако, когда программа начинает становиться слишком сложной, мы инвертируем эти зависимости в противоположную сторону: высокоуровневые функции теперь зависят не от конкретных реализаций, а от интерфейсов, а реализации теперь лежат в своих модулях. **Любая зависимость всегда может быть инвертирована.** В этом и есть достоинство ООП.
* Возможна также реализация инъекции или внедрения зависимости — процесс предоставления внешней зависимости [программному компоненту](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D0%BE%D0%BD%D0%B5%D0%BD%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5). Это ситуация, при которой объект отдаёт заботу о построении требуемых ему зависимостей внешнему, специально предназначенному для этого общему механизму.

# 1.5 Техническое задание

Согласно ГОСТ 34.602-89 «Комплекс стандартов на автоматизированные системы.» Техническое задание является основным документом, определяющим требования и порядок создания (развития или модернизации - далее создания) автоматизированной системы, в соответствии с которым проводится разработка АС и ее приемка при вводе в действие.

В нормативном документе ГОСТ 34.602-89 ТЗ на АС содержит следующие разделы, которые могут быть разделены на подразделы:

1) общие сведения;  
 2) назначение и цели создания системы;  
 3) характеристика объектов автоматизации;

4) требования к системе;  
 5) состав и содержание работ по созданию системы;  
 6) порядок контроля и приемки системы;  
 7) требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие;  
 8) требования к документированию;  
 9) источники разработки.

Техническое задание было разработано на основании требований настоящего ГОСТа.

# 1.5.1 Общие сведения

Задание на выполнение настоящего курсового проекта было выдано по дисциплине «Разработка приложений для Microsoft NET Framework

Наименование: «Разработка веб-сайта на платформе ASP.NET».

Целью разработки является демонстрация умений и навыков составления автоматизированной системы на примере веб-сайта с использованием фреймворка ASP.NET с использованием языка программирования С# в среде программирования Visual Studio 2022, с использованием SQL SERVER Management STUDIO 18 и веб-платформы Blazor.

# 1.5.2 Назначение системы

Данный программный продукт должен обеспечить хранение, изменение и обработку данных согласно теме проекта.

# 1.5.3 Характеристика объекта автоматизации

Система предназначена для решения перечисленных ниже задач:

* Обеспечивать хранение сведений о врачах и услугах и категорий, к которым они относятся;
* Возможность записаться к конкретному врачу или на выбранную услугу;
* Отображение врачей и услуг по выбранным категориям;
* Возможность оставить отзыв о клинике;
* Отображение карты с местонахождением на ней клиники в городе;
* Возможность зарегистрировать аккаунт или войти в существующий;
* Возможность зайти в аккаунт администратора или контент менеджера по определенным данным;
* Корректировать список врачей со стороны администратора – добавлять новых или увольнять;
* Редактировать категории в аккаунте администратора и контент-менеджера;
* Изменять роли и права доступа
* ???????7 контент
* Возможность отображения и функционирования чат-менеджера по технологии SignalR;

# 1.5.4 Требования к системе

На компьютере должны быть установлены:

* операционная система Windows с установленным Microsoft .NET Framework 6.0;
* дополнительно должны быть установлены Visual Studio 2022, SQL SERVER Menagment STUDIO 18.

Visual Studio 2022 может устанавливаться и запускаться в следующих операционных системах:

* Windows 10 версии 1507 и выше: Домашняя, Pro, для образовательных учреждений и Корпоративная (выпуски LTSC и S не поддерживаются).
* Windows Server 2016: Standard и Datacenter Windows 8.1 (с [обновлением 2919355](https://support.microsoft.com/kb/2919355)): Core, Профессиональная и Корпоративная.
* Windows Server 2012 R2 (с [обновлением 2919355](https://support.microsoft.com/kb/2919355)): Essentials, Standard, Datacenter.
* Windows 7 с пакетом обновления 1 (SP1) (с последними обновлениями Windows): Домашняя расширенная, Профессиональная, Корпоративная, Максимальная.
* Процессор с тактовой частотой не ниже 1,8 ГГц. Рекомендуется использовать как минимум двухъядерный процессор.
* 2 ГБ ОЗУ; рекомендуется 4 ГБ ОЗУ (минимум 2,5 ГБ при выполнении на виртуальной машине).
* Место на жестком диске: до 130 ГБ свободного места в зависимости от установленных компонентов, обычно для установки требуется от 20 до 50 ГБ свободного места.
* Скорость жесткого диска: для повышения производительности установите Windows и Visual Studio на твердотельный накопитель (SSD).
* Видеоадаптер с минимальным разрешением 720p (1280 на 720 пикселей); для оптимальной работы Visual Studio рекомендуется разрешение WXGA (1366 на 768 пикселей) или более высокое.

# 1.5.5 Состав и содержание работ по созданию системы

Стадия разработки программного средства включает в себя следующие этапы:

* Получение исходной информации;
* Проектирование алгоритмов;
* Кодирование программы;
* Тестирование и отладка программы;
* Оформление программной документации с описанием пользовательского интерфейса

# 1.5.6 Условия эксплуатации

Данный программный продукт предполагается эксплуатировать в условиях, благоприятных для оператора ЭВМ.

**ГЛАВА 2. ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА**

**ПРОГРАММНОГО РЕШЕНИЯ**

# 2.1 Проектирование и разработка классов

Данный проект был выполнен при помощи фреймворка ASP.NET и использованием MVC – Model View Controller. Паттерн MVC представляет собой разделение данных приложения на три отдельных компонента: модель, представление и контроллер — таким образом, что модификация каждого компонента может осуществляться независимо.

Модель – это компонент, отвечающий за данные, а также определяет структуру приложения.

Представление – отвечает за взаимодействие с пользователем. То есть код компонента определяет внешний вид приложения и способы его использования. Оно имеет расширение cshtml для того, чтобы внутри него была возможность писать код на html для пользователя и код, используя c#, выполняя функционал страницы.

Контроллер – этот компонент отвечает за связь между моделью и представлением. Код компонента определяет, как сайт реагирует на действия пользователя.

В разных проектах с использованием MVC должно происходить статическое хранение данных в модели. В данной стоматологии данные хранятся статически в файлах json, в которые информация была занесена с помощью автоматизированного сбора и структурирования информации с сайтов при помощи программы или сервиса.

Далее должна была разрабатываться база данных, благодаря которой наш сайт будет функционировать. Для этого в ООП есть главный элемент – класс, который включает в себя множество объектов с одинаковыми свойствами, операциями и отношениями. Наш класс для создания таблиц в базе данных называется ApplicationDbContext.

Для того, чтобы база данных имела названия таблиц и столбцы была создана область Domains, хранящая классы с названиями каждой таблицы и столбцы, которые будут созданы и заполнены данными.

Таким образом, у нас получилась пока что пустая база данных – DentistryDb, имеющая такие таблицы, как:

* Service
* Doctor
* Category
* CategoryUslugi
* CategoryMedic
* Reviews

После того, как была создана непосредственно база данных с таблицами, нужно было в неё заносить данные из файлов json, и для этого был использован класс DataSeeder.

Один из важных критериев для дальнейшего удобства работы необходимо было использовать для каждой таблицы - акроним, обозначающий четыре базовые функции, используемые при работе с базами данных – CRUD. Эта аббревиатура обозначает Create, Read, Update, Delete. Для работы с акронимом использовались классы: ServiceSqlRepository, DoctorSqlRepository, CategorySqlRepository, CategoryMedicSqlRepository, CategoryUslugiSqlRepository, ReviewsSqlRepository.

Переходим к заполнению самого главного – логике нашего сайта. Так как наш проект имеет схему MVC, то в первую очередь нам необходима модель, хранящая в себе информацию о наших данных. У нас есть множество классов – моделей, в которых прописаны свойства на все столбцы из базы данных.

Модели:

ServicesModel содержит свойства для таблицы Service. DoctorModel для таблицы Doctor. ReviewsModel для таблицы Review. Также некоторые таблицы содержат сокращенный вид модели для упрощения некоторого функционала – DoctorBriefModel, ServicesBriefModel.

Также на сайте производятся такие функции, как редактирование данных о врачах или услугах, отображение карты с местонахождением, смена ролей при входе в аккаунт. Поэтому для них созданы отдельные модели, хранящие соответствующие свойства – EditDoctorsModel, EditServicesModel, GoogleMapsModel, ChangeRoleViewModel.

После объявления моделей в системе MVC следует создание контроллеров. Данный проект частично был выполнение и использованием технологии Blazor, поэтому мы будем говорить не о всех контроллерах, которые написаны на системе MVC без различных дополнений.

Первый контроллер – для функционирования сайта со стороны администратора AdminController. В нем прописано несколько методов – экшенов, отвечающих за удаление и добавление врача, редактирование категорий – изменение, добавление и удаление.

Сюда же относится контроллер на выполнение логики контент-менеджера – ContentManagerController. Как и в контроллере администратора выполняется логика редактирования категорий, возможность удалить отзыв на странице клиники, добавить новую услугу или нового врача.

Следующий контроллер, который не использовал технологию Blazor - это GoogleMapController, отвечающий за карту на сайте. В нем создается список данных о названии улицы и геолокации и возвращает json.

И самый главный - BlazorController, отвечающий за остальную логику сайта, который переносит на эту логику на представление.

Исходя из контроллера, фреймворк создает представление, в котором будет реализовываться логика для пользователя. Точно также, как и в контроллере, большую часть функционала на сайте выполняет blazor, поэтому мы начнем с той области, которая за него не отвечает.

Первое - папка Admin, содержащая несколько представлений на то, чтобы: редактировать категории, увольнять врача и редактировать список врачей.

Также и создаются представления для контент-менеджера для того, чтобы менеджер мог редактировать услуги или врачей и удалять отзывы.

Следующее представление – GoogleMap.Index содержит логику выполнения карты с местонахождением. В этом файле вся логика пишется на языке JavaScript. Здесь устанавливаются основные координаты, общие параметры отображения карты. Затем происходит встраивание гугл-карты в элемент на странице и получение объекта карты и получаем данные.

Также на всех страницах закреплен базовый шаблон, в котором есть шапка сайта, цвета и стили, и этот шаблон создается в странице \_Layout – Blazor, которая уже привязана к каждой странице сайта.

Перейдем к рассмотрению самого главного - представления, на которое нас переносят специальные методы в контроллере BlazorController, которые называются экшенами. Это IndexMain, IndexDoctor и Index в папке Blazor.

Структура blazor устроена таким образом, что в представлении мы можем написать метод RenderComponentAsync, который будет перенаправлять на страницу razor, в которой и написана вся логика для этой страницы.

В этой стоматологии страницы razor, которые отвечают за всю логику сайта на стороне blazor, находятся в отдельной папке – BlazorComponents. Начнем рассматривать все эти страницы.

Blazor состоит из компонентов, поэтому для удобного разделения частей сайта, здесь использована привязка нужных страниц с выполнением функционала на страницу, которая уже отображает их.

В первую очередь рассмотрим ту, что отвечает за отображение начальной страницы на нашем сайте, и, к которой привязаны некоторые компоненты. Это AppleMainComponent, и к ней же привязан компонент razor - BlazorMessageComponent.

Также есть компоненты, отвечающие за список услуг, врачей и просмотр их более детально – BlazorTableServicesComponent, BlazorDocTableComponent, BlazorDialogDocComponent, ServiceModalComponent. А главные компоненты, на которые они подкрепляются – это AppleComponent, MainDocComponent.

Для запуска всего приложения в Visual Studio есть специальный класс – Program. Но в нашем случае, он еще и отправляет на главный класс – Startup, который производит конфигурацию приложения, настраивает сервисы, которые приложение будет использовать, устанавливает компоненты для обработки запроса.

**2.2 Проектирование и разработка пользовательского интерфейса**

# 2.2.1 Главная страница

Начальная страница, с которой открывается сайт (рис.10)

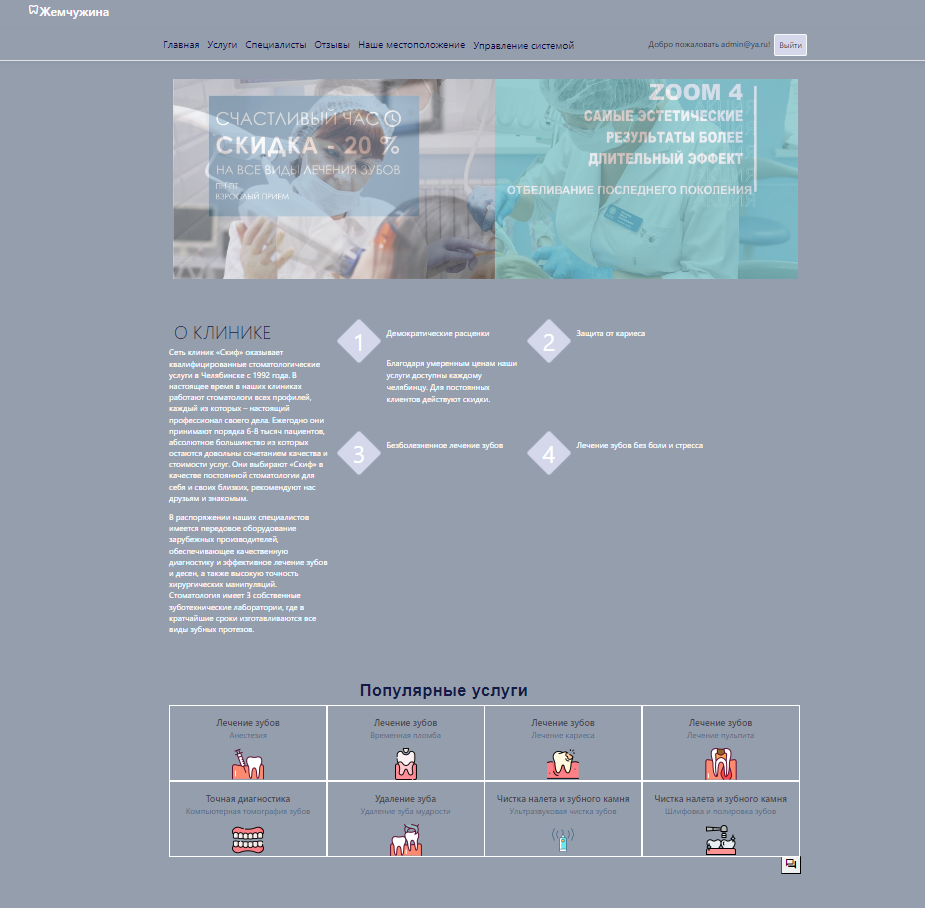


Рисунок - Главная страница

Здесь для пользователя показана информация о клинике, а именно:

* Слайдер, который имеет анимацию, и выполнен при помощи технологии blazor. На этом слайдере показаны картинки с различными акциями стоматологии;
* Описание клиники;
* Таблица с популярными услугами клиники;
* Чат-менеджер, которому в любое время можно задать интересующий вопрос (рис.11).

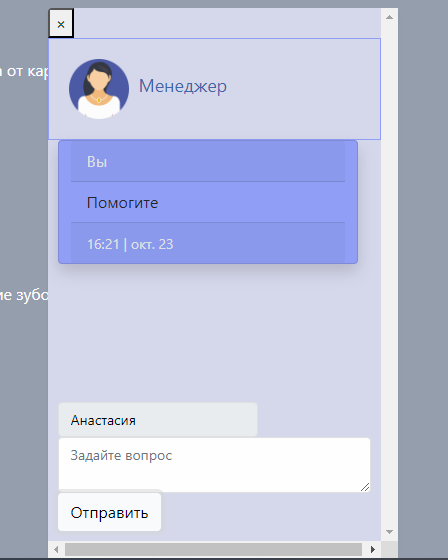


Рисунок - Чат-менеджер

# 2.2. 2 Страница со списком услуг

Пользователь может ознакомиться с услугами и ценами клиники по выбранной категории (рис.12). Также можно записаться на выбранную услугу.

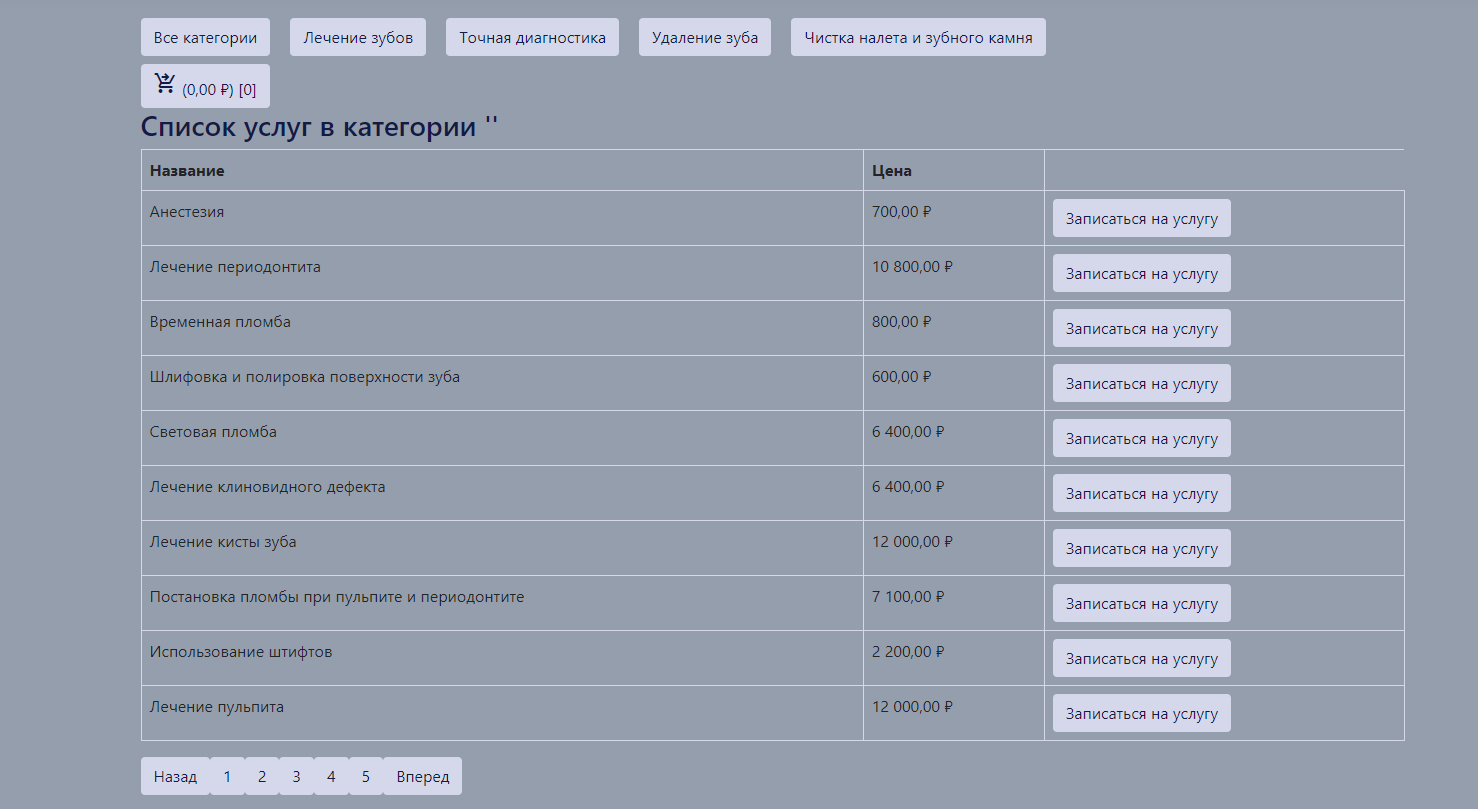


Рисунок - Услуги

# 2.2. 3 Страница со списком специалистов

Пользователь может ознакомиться со специалистами клиники (рис.13).

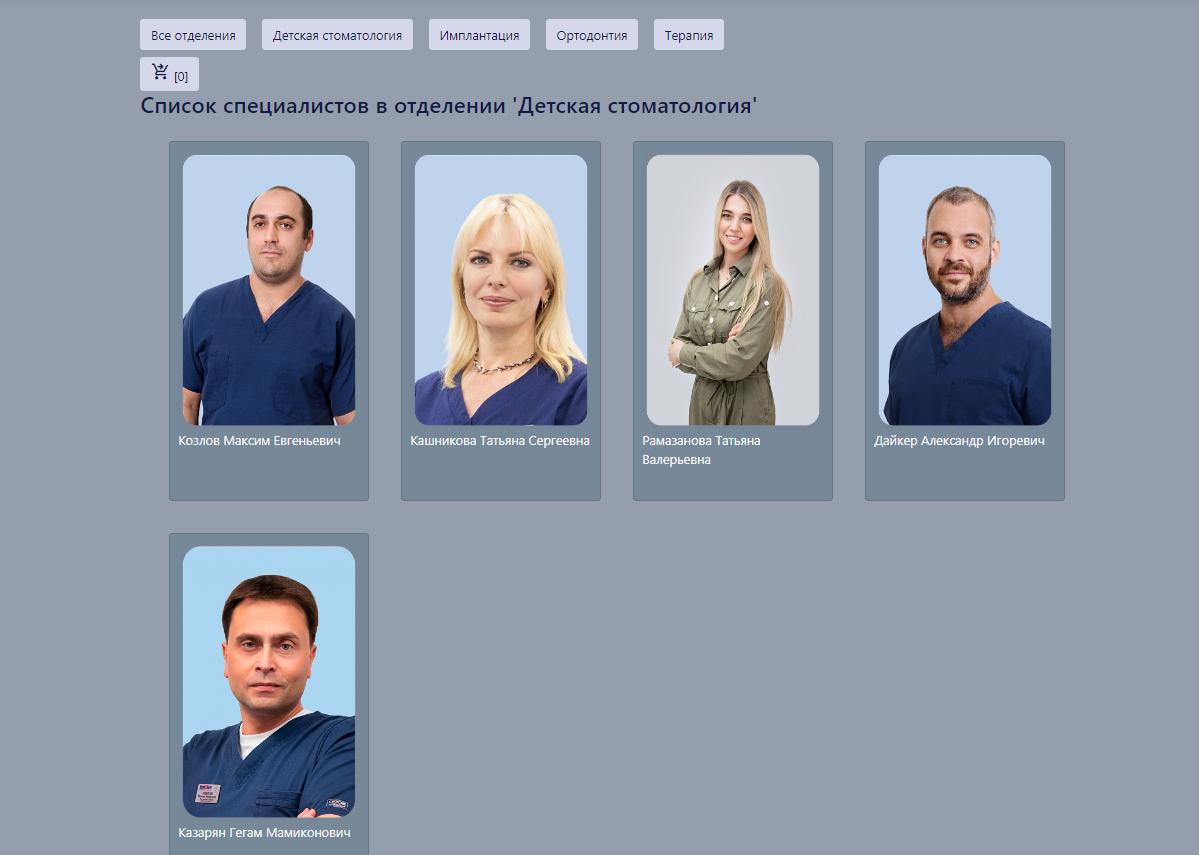


Рисунок - Специалисты

Если есть необходимость – можно просмотреть более детальную информацию о каждом враче. Откроется модальное окно с именем врача, описанием и фото (рис.14). И здесь же можно записаться к этому врачу.

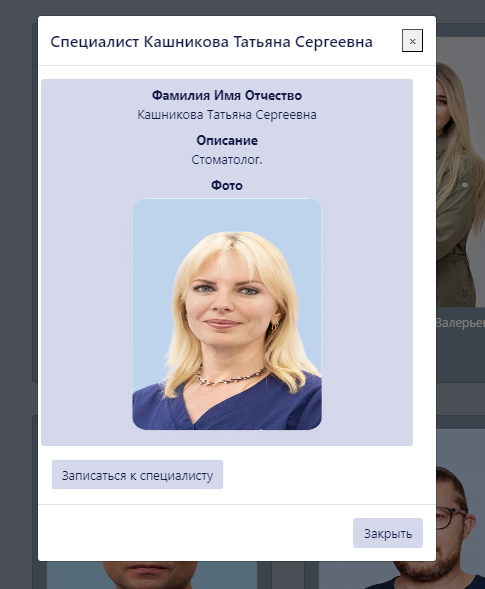


Рисунок - Модальное окно специалиста

# 2.2.4 Страница отзывов

В данном месте пользователь может оставить отзыв о клинике (рис. 15)

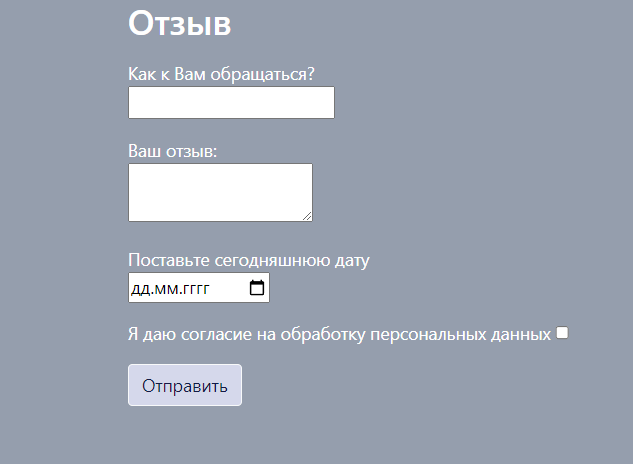


Рисунок - Отзыв

# 2.2.5 Местонахождение клиники

Для удобства, на сайте есть информация о местонахождении клиники (рис.16)

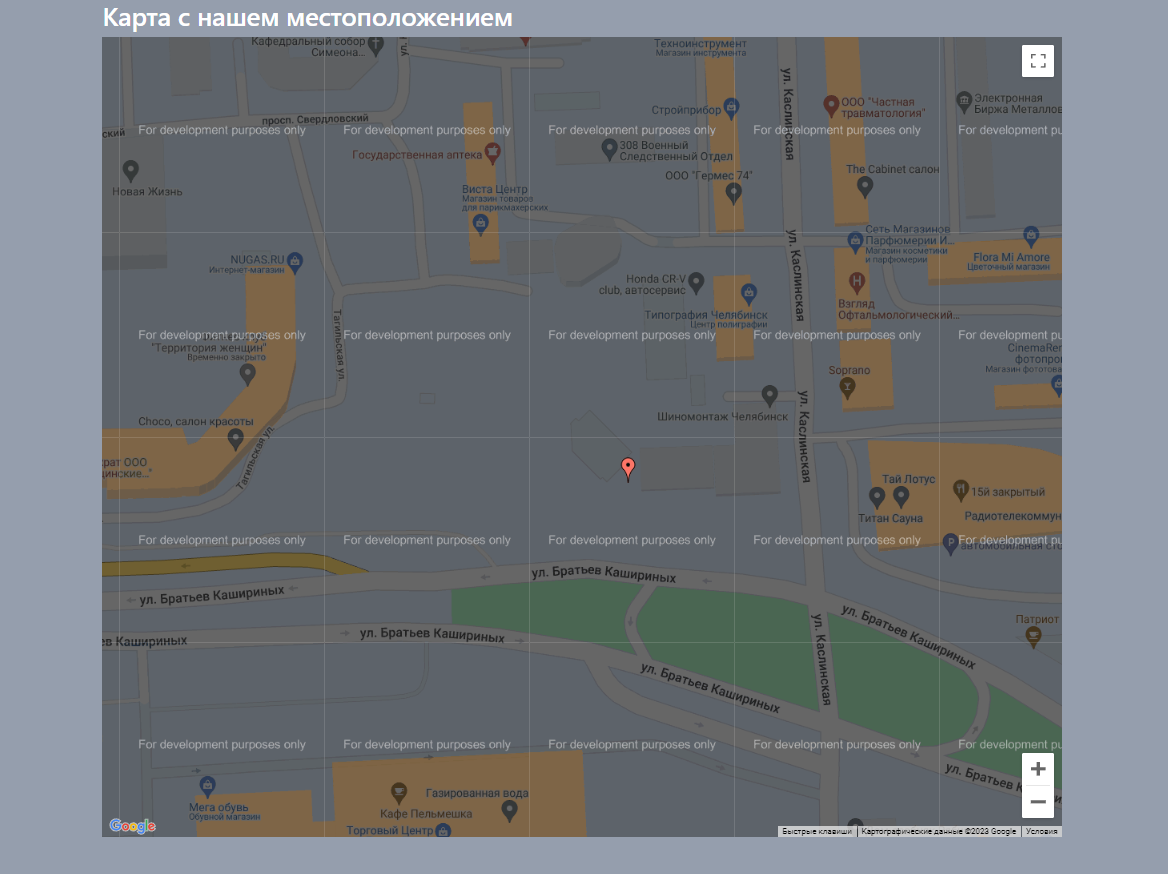


Рисунок - Карта

# 2.2.6 Корзина пациента

В данном проекте осуществлена функция добавления и хранения талонов в корзине к записанным врачам и услугам (рис. 17).

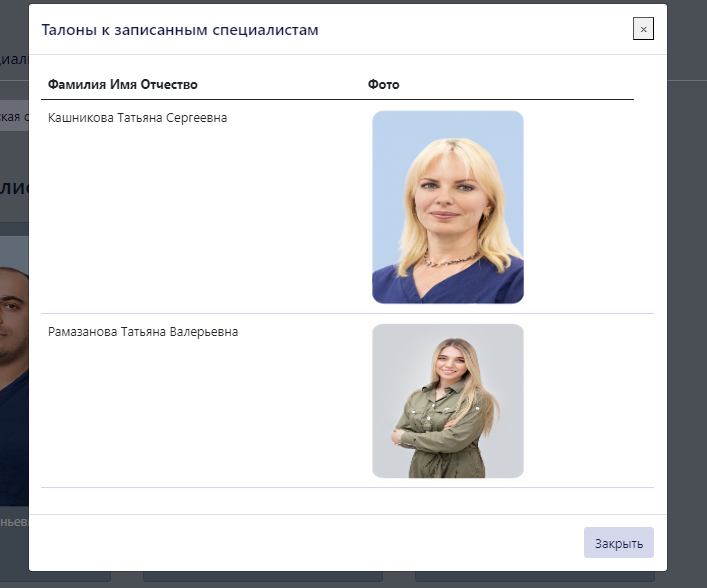


Рисунок - Корзина

# 2.2.7 Главная страница администратора

Данная страница содержит кнопки перехода на определенные возможности администратора (рис. 18)

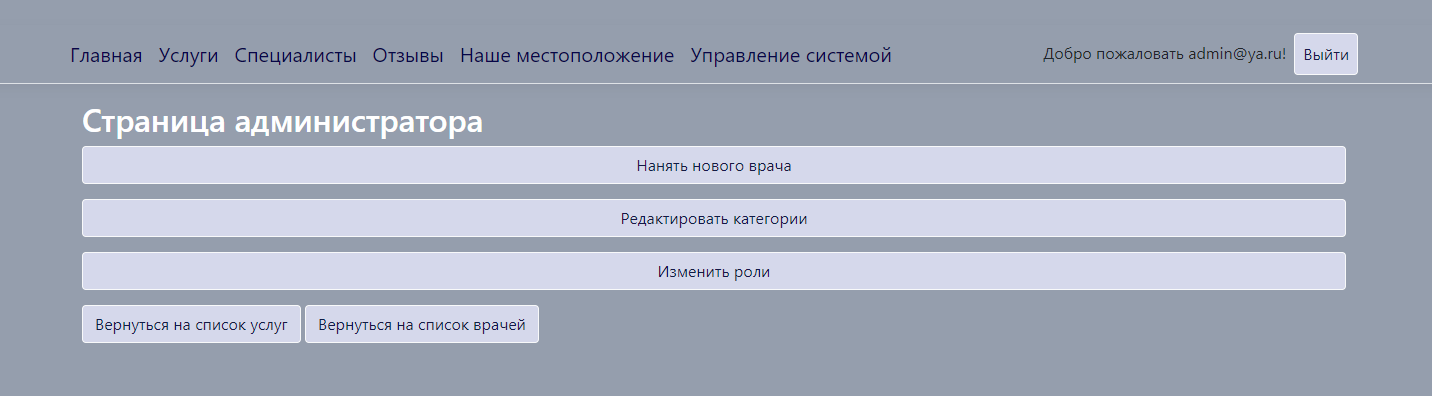


Рисунок - администратор

# 2.2.8 Нанять врача

Здесь администратор может нанять нового врача в клинику, заполнив о нем необходимые данные и прикрепить фото (рис. 19)

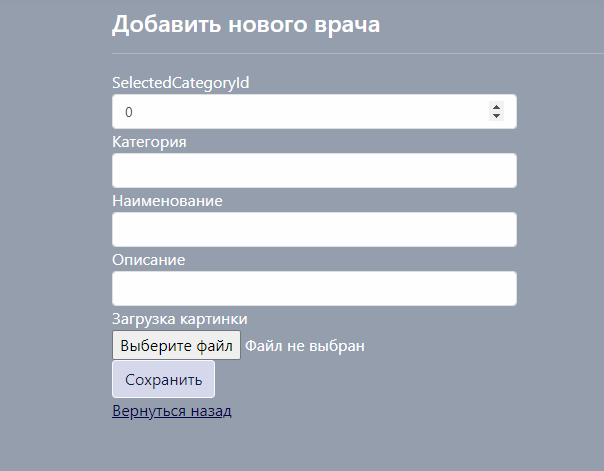


Рисунок - нанять врача

# 2.2.9 Редактирование категорий

Здесь администратор может редактировать категории – добавлять новые, изменять или удалять старые (рис.20). Услуги и специалисты относятся к какой-либо категории, поэтому они очень важны.

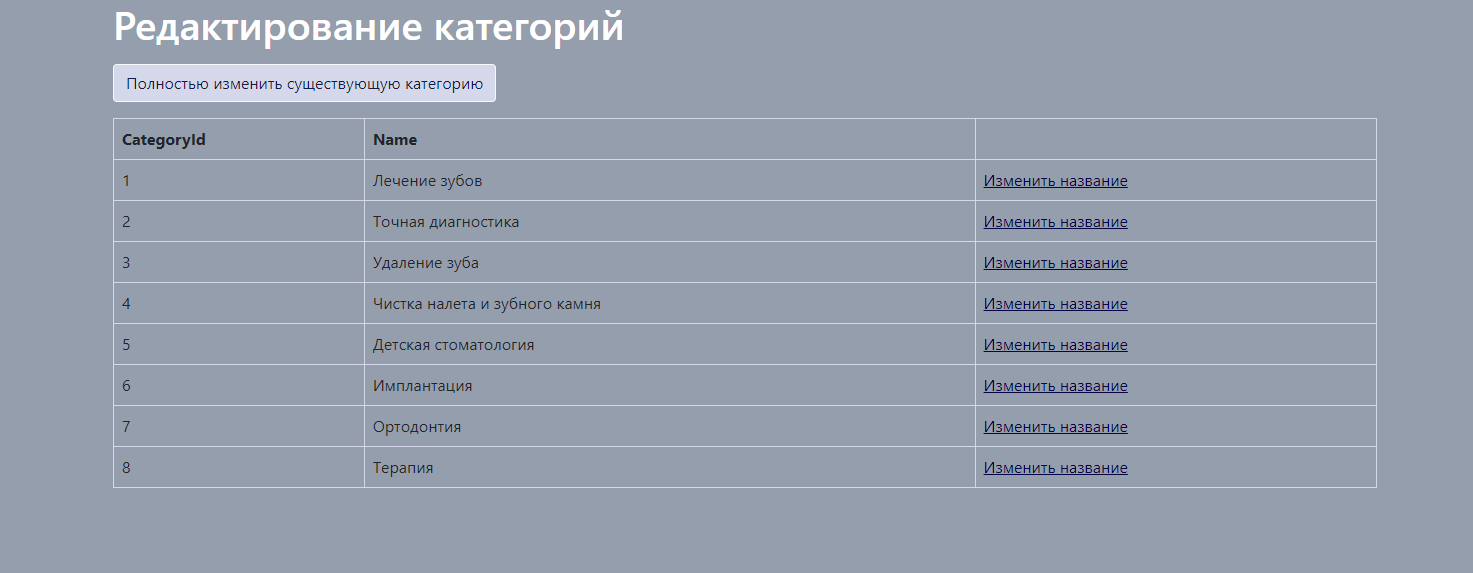


Рисунок - категории

# 2.2.10 Роли

Самый важный функционал у администратора – это изменение ролей (рис. 21).

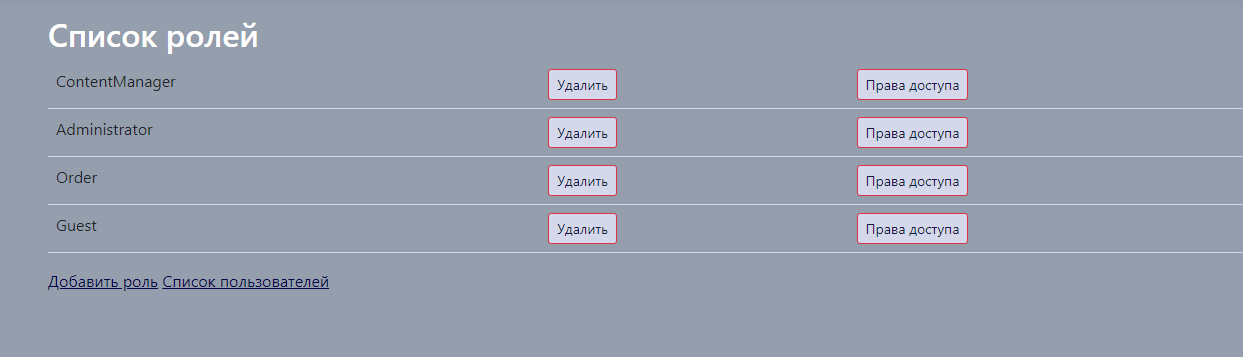


Рисунок - роли

Он может управлять ролями – удалить существующую, добавить новую или изменить права доступа для каждой роли (рис. 22).

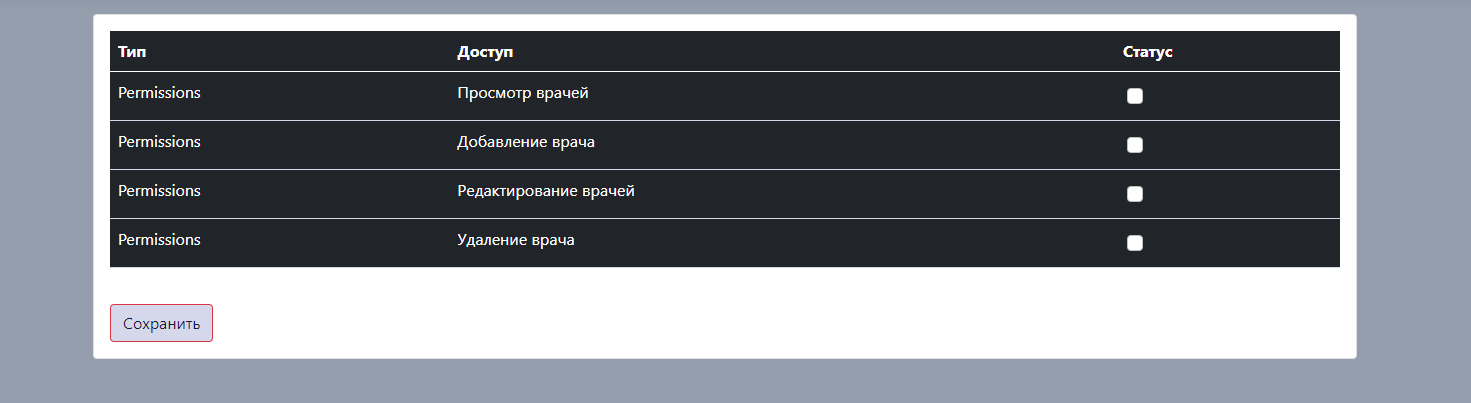


Рисунок - права доступа

# 2.3 Описание и разработка базы данных

Для эффективной работы была разработана база данных. Стоматология реализована не только в услугах и специалистах, но и ролях возможности наличия аккаунта для обычного пользователя или работника клиники. Для этого были автоматически созданы таблицы с ролями (рис. 23).

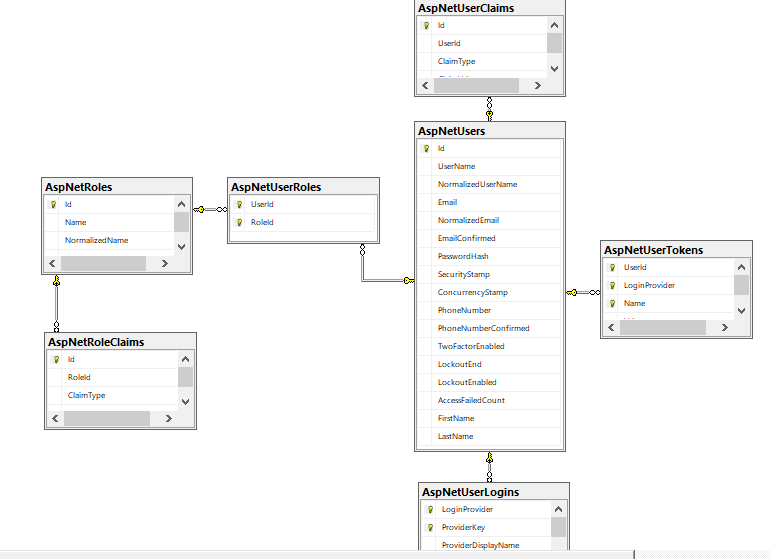


Рисунок - таблицы ролей

Кроме вышеперечисленных таблиц, были спроектированы те, которые хранят в себе все данные проекта (рис. 24).

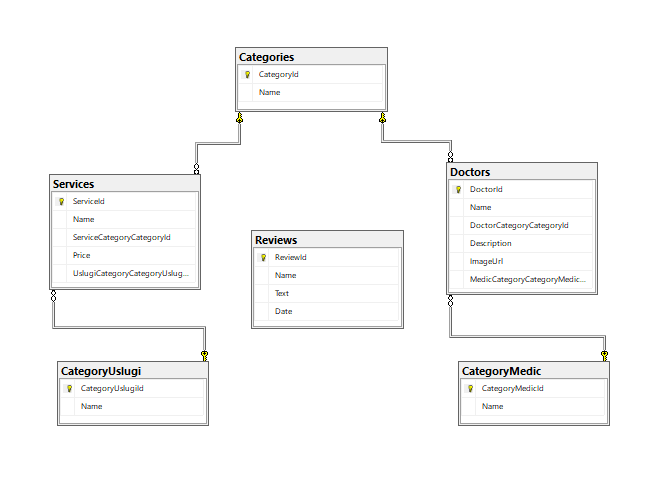


Рисунок - база данных

Эти таблицы содержат в себе следующие параметры:

Categories имеет в себе 2 поля – идентификатор и название.

От нее исходят по вторичному ключу еще две таблицы:

Services, в которой есть идентификатор, название, идентификатор вторичного ключа категорий, цена и еще один идентификатор вторичного ключа к таблице категории-услуги.

И Doctors, имеющая идентификатор, название, также два идентификатора к таблицам категории и категории-врачи, описание и картинка.

Следующие две таблицы, к которым привязаны вышеперечисленные – CategoryUslugi и CategoryMedic, содержащие только идентификаторы и названия.

Последняя таблица, никак не связанная с остальными и отвечающая за отзывы на сайте, которая не статично хранит в себе данные, а они в не записываются непосредственно с сайта в момент, когда пользователь оставляет отзыв, – это Reviews, и в ней хранятся такие столбцы, как идентификатор, имя того, кто оставляет отзыв, текст и дата.

Данная база позволяет хранить данные в отдельных таблицах. Таблицы взаимодействуют друг с другом путём установления в них отношений.

Отношения позволяют хранить данные в разных таблицах, связывая их отношениями данные из одной таблицы передаются в другую.

# 2.4 Краткое руководство пользователя по эксплуатации системы

* 1. Открыть файл с именем KursovayaBlazorNet6
  2. На открывшемся в браузере сайте можно переходить между страницами.

# ЗАКЛЮЧЕНИЕ

Работая над данной курсовой работой, мною были изучены и закреплены основы объектно-ориентированного подхода к программированию на языке С# с использованием инструментальной среды MS Visual Studio 2022, использования фреймворка ASP.NET с использованием модели MVC, текстового языка программирования – JavaScript, использования Json, технологии Blazor, библиотеки SignalR.Получилось достигнуть главной цели, а именно разработка и автоматизированное использование сайта – стоматологии. В ходе работы, также были достигнуты основные задачи, которые были представлены поэтапно. Укрепились навыки постановки задач, разработки алгоритмов и их решений, разработки технического задания, разработки и описания пользовательского интерфейса.

Исходя из всего вышеперечисленного, можно сказать, что проектирование и разработка проектов на платформе ASP.NET может достаточно хорошо заменить и автоматизировать работу разных подобных платформ.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. ГОСТ 34.602-89. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы [Электронный ресурс] //Электронный фонд правовых и нормативно-технических документов. – Режим доступа: <https://docs.cntd.ru/classifications>

2. Полное руководство по языку программирования С# 10 и платформе .NET 6 [Электронный ресурс] //Metanit.com. Сайт о программировании – Режим доступа <https://metanit.com/sharp/tutorial/>

3. Объектно-ориентированное программирование (C#) [Электронный ресурс] //Microsoft. Документация – Режим доступа: <https://docs.microsoft.com/ru-ru/dotnet/csharp/fundamentals/tutorials/oop>

4. Объектно-ориентированное программирование [Электронный ресурс] //Код. Журнал Яндекс Практикума. – Режим доступа: <https://thecode.media/objective/>

5. Введение в ООП с примерами на С# [Электронный ресурс] // Tproger.– Режим доступа: <https://tproger.ru> 6. Объектно-ориентированное программирование [Электронный ресурс] //Skilbox Media – Режим доступа: <https://skillbox.ru/media/>

7. Объектно-ориентированное программирование [Электронный ресурс] //Библиофонд – Режим доступа: <https://www.bibliofond.ru/view.aspx?id=700569>

8. Предметная область и ee моделирование [Электронный ресурс] //РРепетитор. – Режим доступа: <file:///C:/Users/Home/Desktop/курсовая/Предметная%20область>

9. Предметная область [Электронный ресурс] //Википедия. – Режим доступа: <https://ru.wikipedia.org/wiki/Предметная_область>

10. Класс (программирование) [Электронный ресурс] //Википедия. – Режим доступа: <https://ru.wikipedia.org/wiki/Класс_(программирование)>

11. Объект (программирование) [Электронный ресурс] //Википедия. – Режим доступа: <https://ru.wikipedia.org/wiki/Объект_(программирование)>

12. Модификаторы доступа (программирование) [Электронный ресурс] //Википедия. – Режим доступа: <https://translated.turbopages.org/proxy_u/en-ru.ru.f086b740-628229d2-ecb19e75-74722d776562/https/en.wikipedia.org/wiki/Access_modifiers>

13. Абстрактный класс (программирование) [Электронный ресурс] //Википедия. – Режим доступа: <https://ru.wikipedia.org/wiki/Абстрактный_класс>

14. Интерфейс (программирование) [Электронный ресурс] //Википедия. – Режим доступа: <https://ru.wikipedia.org/wiki/Интерфейс>

15. Модульное программирование [Электронный ресурс] //Википедия. – Режим доступа: <https://ru.wikipedia.org/wiki/Модульное_программированиe>

16. Связи между таблицами базы данных [Электронный ресурс] //Хабр. – Режим доступа: <https://habr.com/ru/post/488054/>

17. NET для начинающих. Что такое среда .NET и как она работает? [Электронный ресурс] //Хабр – Режим доступа: <https://habr.com/ru/sandbox/13849/>

18. Новые возможности Visual Studio 2017. [Электронный ресурс] //Microsoft. Документация – Режим доступа: <https://docs.microsoft.com/ru-ru/visualstudio/ide/whats-new-visual-studio-2017?view=vs-2017>

19. Что такое SQL Server Management Studio (SSMS). [Электронный ресурс] //Microsoft. Документация – Режим доступа: <https://docs.microsoft.com/ru-RU/sql/ssms/sql-server-management-studio-ssms?view=sql-server-ver15&viewFallbackFrom=sql-server>