Утверждаю

Руководитель предприятия

(Ф.И.О.)

(подпись, печать предприятия)

«     »   апреля      2023 г.

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет   Информационных Технологий

Кафедра     Программной инженерии

Специальность 1-40 01 01 Программное обеспечение информационных технологий

ОТЧЕТ

по производственной преддипломной практике

на   ООО «СофтТеко» с 20 марта 2023г. по 16 апреля 2023 г.

(наименование предприятия, сроки практики)

Исполнитель

Студент   4   курса     5     группы                            Савченко В. Ю.

(подпись, дата) (Ф.И.О.)

Руководитель практики

от предприятия

Программист-инженер                                      Пликус Д. Г.

(должность, печать предприятия) (подпись, дата) (Ф.И.О.)

Руководитель практики

от университета

Ассистент                                                                     Пахолка А. С.

(должность, уч. звание) (подпись, дата) (Ф.И.О.)

Отчет защищен с оценкой\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Минск 2023

**Реферат**

Отчет по производственной преддипломной практике состоит из 25 страниц, 10 рисунков, 0 таблиц, 0 приложений, 4 использованных источников.

WEB-ПРИЛОЖЕНИЕ, AdYou, C#, .Net Core, Angular, SqLite, DOCKER, TYPESCRIPT, AXIOS, Ef Core

Объектом производственной преддипломной практики является веб-приложение.

Цель производственной преддипломной практики – разработка веб-приложения для предоставления рекламы, которое позволяет пользователям находить более выгодные предложения, а компаниям рекламные услуги, чтобы увеличить число их клиентов.

В разработке проекта был использован язык программирования C#, JavaScript фреймворк Angular, контейнеризатор Docker, среда запуска и тестирования на базе операционной системы Windows 10.

Отчет состоит из введения, двух разделов, заключения и списка литературы.

В первом разделе производится аналитический литературы по тематике производственной преддипломной практики, а также постановка задачи.

Второй раздел посвящен проектированию веб-приложения.

В заключении представлены результаты производственной преддипломной практики.

**Abstract**

The report on industrial undergraduate practice consists of 25 pages, 10 figures, 0 tables, 0 applications, 4 references.

WEB APP, AdYou, C#, .Net Core, Angular, SqLite, DOCKER, TYPESCRIPT, AXIOS, Ef Core

The object of the industrial undergraduate practice is a web application. The aim of the internship is to develop a web-based advertising application that allows users to find better deals and companies advertising services to increase the number of their customers. In the development of the project, the C# programming language, the Angular JavaScript framework, the Docker containerizer, the launch and testing environment based on the Windows 10 operating system were used. The report consists of an introduction, two sections, a conclusion and a list of references. In the first section, analytical literature is produced on the subject of industrial undergraduate practice, as well as the formulation of the problem. The second section is about designing a web application.

In conclusion, the results of the industrial pre-diploma practice are presented.

**Содержание**

[Введение 5](#_Toc132564201)

[1 Постановка задачи и обзор аналогичных решений 6](#_Toc132564202)

[1.1. Постановка задачи 6](#_Toc132564203)

[1.2. Обзор аналогов 7](#_Toc132564204)

[1.2.1 SLIVKI.BY 7](#_Toc132564205)

[1.2.2 Promokods.BY 10](#_Toc132564206)

[1.3. Выводы по разделу 12](#_Toc132564207)

[2 Проектирование 13](#_Toc132564208)

[2.1. Выбор инструментов разработки 13](#_Toc132564209)

[2.2. Диаграмма вариантов использования 15](#_Toc132564210)

[2.3. Архитектура 17](#_Toc132564211)

[2.4. Разработка модели базы данных 18](#_Toc132564212)

[2.5. Выводы по разделу 20](#_Toc132564213)

[Заключение 21](#_Toc132564214)

[Список использованной литературы 22](#_Toc132564215)

# Введение

ООО «СофтТеко» – международная компания-разработчик программного обеспечения на заказ с офисами в Минске, Гомеле, Витебске, Киеве, Варшаве, Вильнюсе и Москве. История SoftTeco началась в 2008 году. Сегодня мы сильная команда из более чем 480 ИТ- инженеров, профессионалов в разработке ПО, тестировании, бизнес-анализе, менеджменте и продажах.

Компания создает и предоставляем ориентированные на результат решения и услуги, идет в ногу с передовыми инновациями и поддерживаем долгосрочные отношения, которые приносят пользу клиентам в долгосрочной перспективе.

В 2012 году «СофтТеко» стала резидентом Парка высоких технологий и получила 3-е место в проводимом в то время конкурсе Best It company to work for in Belarus в категории «компании 11−50 человек». Компания имеет офисы в Минске, Гомеле, Витебске, Киеве, Варшаве, Вильнюсе и Москве. Старшие разработчики полного стека, умеющие использовать множество технологий и фреймворков, имеют более чем семилетний опыт работы и работают вместе со многими другими специалистами среднего и младшего звена, что позволяет молодым специалистам быстро получить опыт коммерческой разработки.

Целью производственной преддипломной практики является изучение стандарта БГТУ по дипломному проектированию. Необходимо провести анализ информационных источников (системы, методы, алгоритмы, программные решения, технологии программирования) по теме дипломного проекта. Также нужно провести анализ требований, разработать функциональные требования и архитектуру проектируемого приложения. И выполнить реализацию приложения в соответствии с индивидуальным заданием.

Итогом же данной практики будет полностью функционирующее и логически завершенное веб-приложение для предоставления рекламных услуг с соответствующим отчётом.

1. Постановка задачи и обзор аналогичных решений
   1. Постановка задачи

С развитием технологий рекламная индустрия также развивается, причем весьма быстрыми темпами. Появление новых технологий повышает эффективность рекламных кампаний, а также увеличивает аудиторию, которая может быть достигнута. Одним из наиболее важных аспектов этой индустрии является предоставление информации о продуктах и услугах. Ежедневно миллионы людей ищут информацию о товарах и услугах в Интернете, и реклама является одним из наиболее эффективных способов достижения потенциальных клиентов.

Цель данного проекта - разработать веб-приложение для предоставления рекламы, которое будет предоставлять рекламные услуги различным компаниям. Это приложение поможет улучшить эффективность рекламных услуг, а также увеличить количество людей, которые могут быть достигнуты. Благодаря возможностям Интернета, наше приложение будет иметь широкую аудиторию, что позволит компаниям достигать большего количества потенциальных клиентов.

Исходя из поставленной цели, необходимо реализовать серверную и клиентскую часть приложения.

Клиентская часть должна:

* обладать удобным интерфейсом с приятным дизайном;
* проводить начальную проверку некоторых данных.

Серверная часть должна:

* обеспечивать доступ к данным;
* обрабатывать запросы и формировать на них ответ.

Для выполнения всех вышеперечисленных условий был выбран следующий набор технологий:

* язык программирования C# и фреймворк .Net Core для серверной части приложения;
* JavaScript фреймворк Angular для клиентской части приложения;
* Sqlite для хранения данных;
* программное обеспечение Docker и Docker Compose для автоматизации развёртывания и управления приложением.

Данное приложение для предоставления рекламы будет полезно как для людей, так и для компаний. Для людей оно будет полезно тем, что позволит им получать информацию о товарах и услугах, которые им могут быть интересны, а также получать различные скидки и предложения от компаний. Для компаний же оно будет полезно тем, что позволит им достигать целевой аудитории и повышать узнаваемость своего бренда, а также увеличивать продажи своих товаров и услуг. Кроме того, приложение позволит компаниям оценивать эффективность своей рекламной кампании и анализировать поведение потребителей.

* 1. Обзор аналогов

В ходе подготовки разработки приложения прежде всего стоит проанализировать уже существующие аналоги на рынке. Среди них были выбраны 2 самых популярных и востребованных.

Требования к аналогам выдвигаются следующие:

– удобство использования: пользователь должен без каких-либо проблем разобраться в функционале, предоставляемом приложением;

– доступность: использование приложения при доступе к интернету только путем использования браузера, без скачивания дополнительных программ и утилит;

– юзабилити: понятное и доступное расположение всех основных функций приложения, а также нативно понятное управление и использование функционала приложения;

### SLIVKI.BY

Первым аналогом для обзора был выбран электронный ресурс «SLIVKI.BY» [1], представлен на рисунке 1.1.

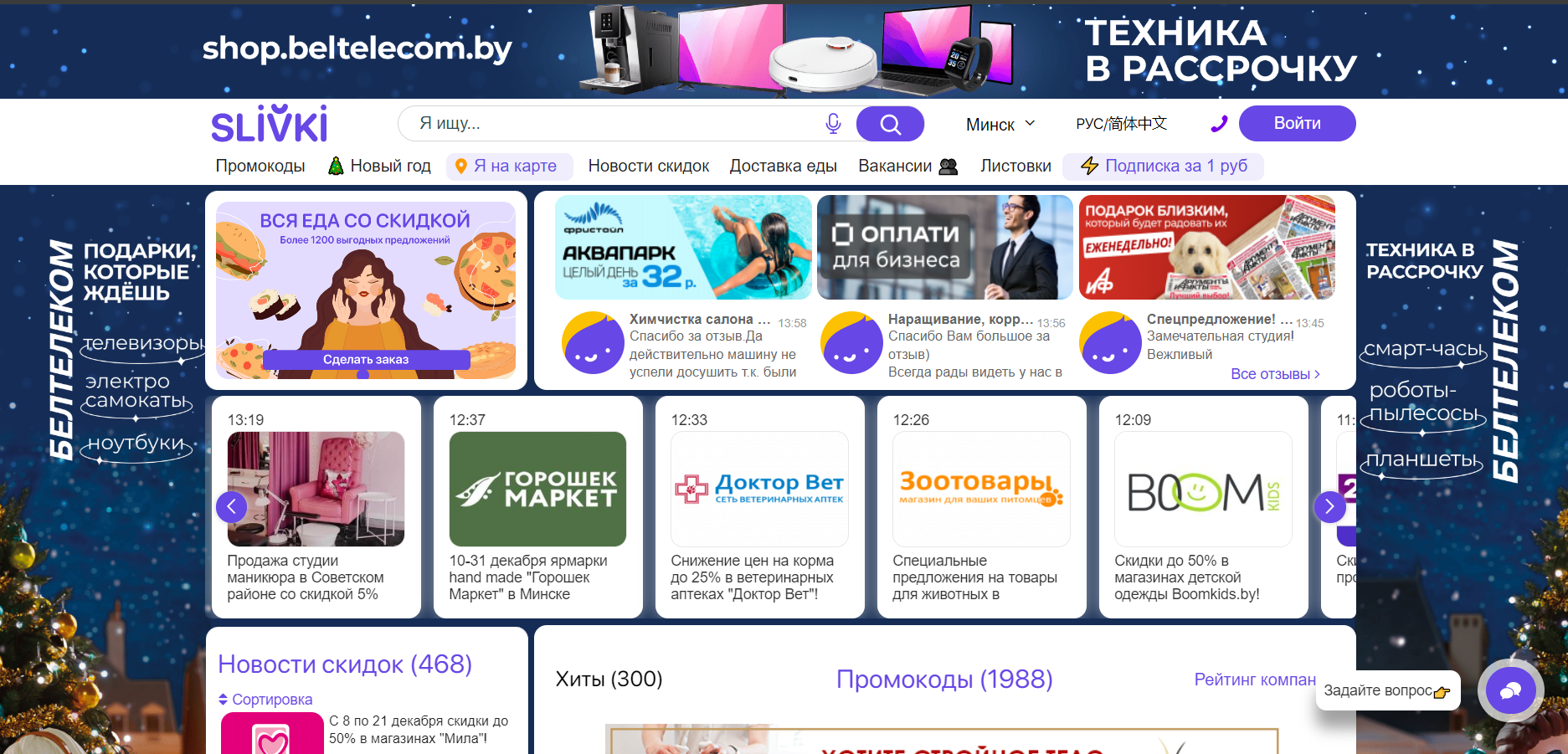


Рисунок 1.1 – Slivki.BY

Slivki.BY­ – это комплексная платформа по рекламе и продвижению брендов и торговых марок в Беларуси. С ее помощью клиенты могут получить качественное рекламное обслуживание, учитывающее все их пожелания и требования. Она предлагает широкий спектр услуг по продвижению товаров и услуг, а также созданию и продвижению идей от имени компании.

Кроме того, Slivki.BY является одним из крупнейших дисконтных сайтов в Беларуси, предоставляющим купоны и промокоды на скидки. Чтобы создать акцию, компании могут заполнить специальную форму на сайте. Благодаря такому подходу Slivki.BY может предоставить выгодные условия для сотрудничества как клиентам, так и компаниям. В целом, платформа может значительно облегчить продвижение бизнеса и помочь компаниям привлечь новых клиентов и повысить узнаваемость своего бренда на рынке.

Начальная страница сайта с данным приложением позволяет нам увидеть основную структуру приложения и методы работы с ним.

Начальная страница представлена на рисунке 1.2.

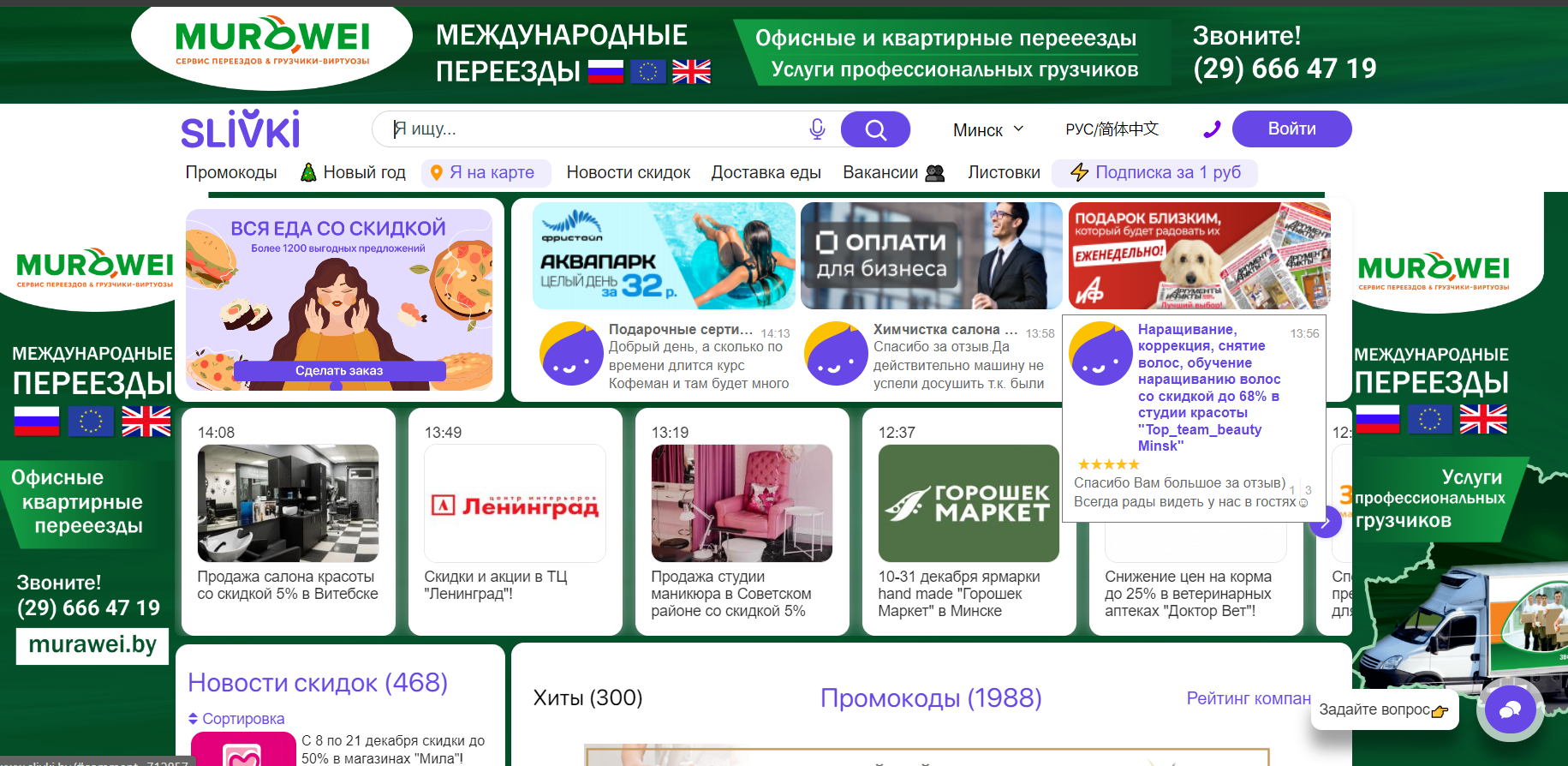


Рисунок 1.2 – Главная страница Slivki.BY

При поиске товара или услуги на сайте может возникнуть ситуация, когда пользователь не знает, где искать нужный ему товар. В таком случае на помощь приходят категории, которые сгруппированы по общим признакам. Это позволяет пользователям сузить область поиска и быстрее найти нужный товар или услугу.

При нажатии на категорию открывается страница с предложениями, которые относятся к этой категории. Это может быть как одна страница с товарами, так и несколько страниц, где пользователь может выбрать нужный ему товар или услугу. Также на страницах с товарами или услугами можно увидеть фильтры, которые помогут сузить область поиска, если предложений слишком много.

Однако, если пользователь не может найти нужный товар или услугу среди категорий, то он может воспользоваться строкой поиска. С ее помощью можно найти товар или услугу по ключевому слову или фразе.

Таким образом, наличие категорий и строки поиска на сайте. Поиск представлен на рисунке 1.3.

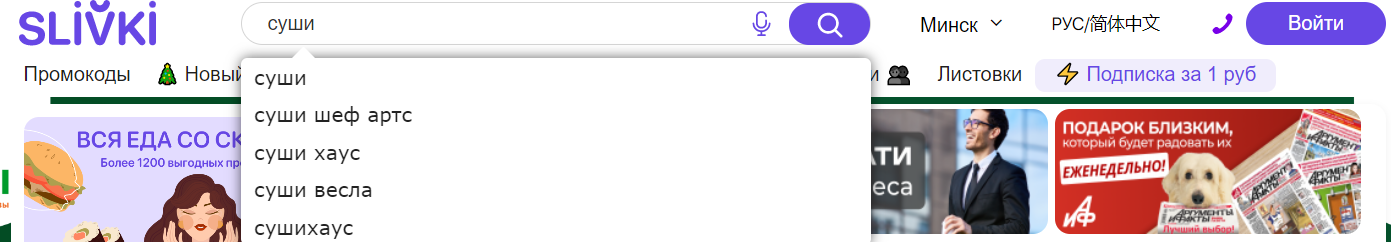


Рисунок 1.3. – Использование функции поиска в Slivki.BY

После того, как пользователь нашёл интересующий его товар, он может перейти в него, кликнув по нему для просмотра более детальной информации.

Страница с детальной информацией представлена на рисунке 1.4.

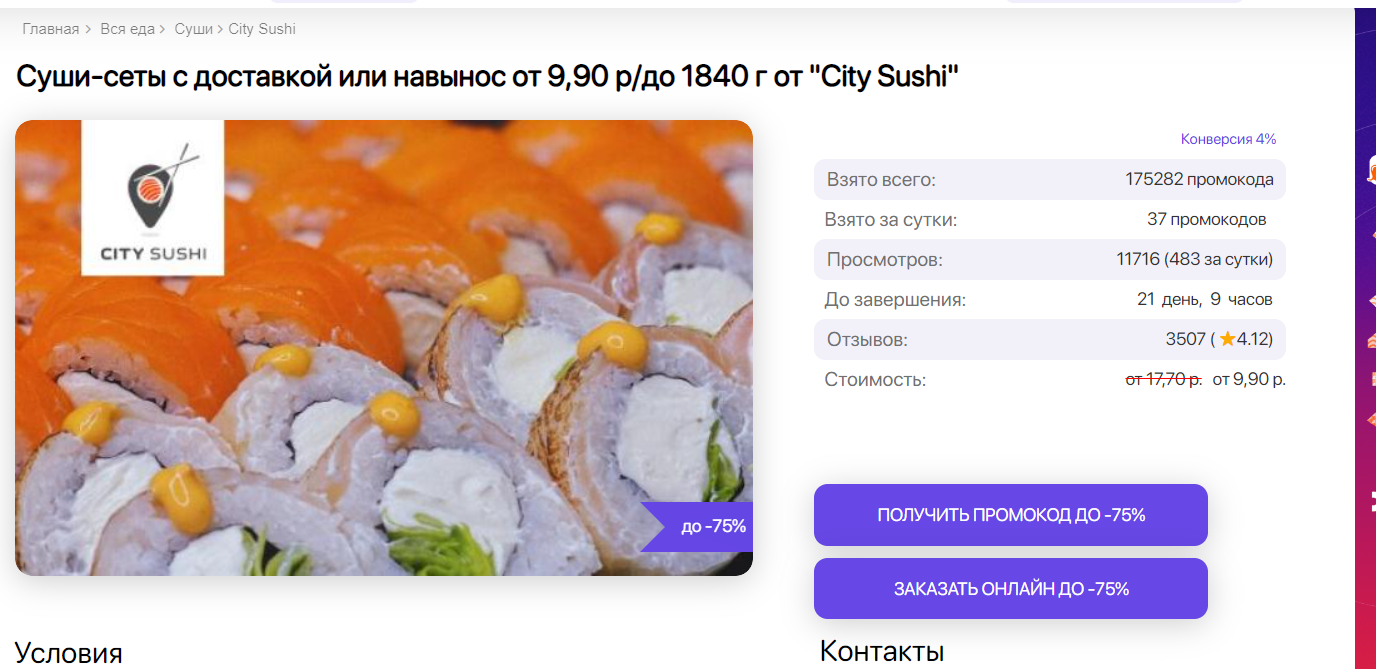


Рисунок 1.4 – Страница с детальной информацией

Если пользователь решает приобрести промокод, то он может нажать на кнопку "Купить" и выбрать удобный способ оплаты. После этого промокод появится в личном кабинете пользователя. Для использования промокода нужно перейти на сайт магазина и ввести его в соответствующее поле при оформлении заказа.

При этом стоимость заказа автоматически уменьшится на сумму скидки, указанную в промокоде. Если у пользователя возникли вопросы по использованию промокода или возникли проблемы с его активацией, он может обратиться в службу поддержки сайта.

Команда специалистов всегда готова помочь в решении любых вопросов, связанных с использованием промокодов. Окно с возможностью получения промокода представлено на рисунке 1.5.

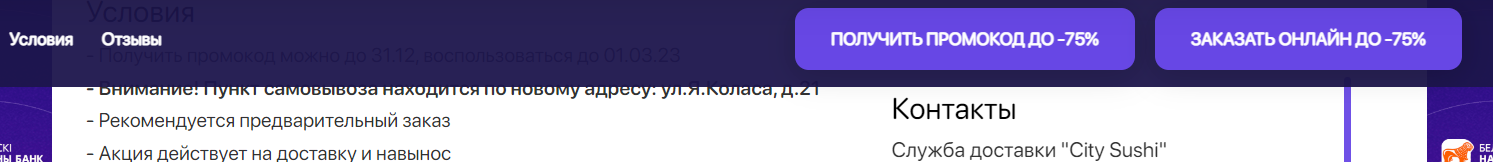


Рисунок 1.5 – Окно с возможностью получения промокода

После нажатия на кнопку «Оплатить», пользователю будет предложено выбрать способ оплаты, после чего промокод будет автоматически выслан на аккаунт пользователя.

Окно покупки промокода представлено на рисунке 1.6.

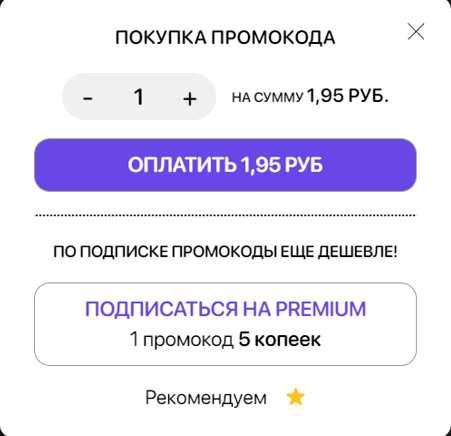


Рисунок 1.6. – Окно покупки промокода

### Promokods.BY

Promokods.BY[2] – онлайн-сервис, предназначенный для помощи гражданам Беларуси совершать выгодные покупки в отечественных и зарубежных интернет-магазинах и сервисах. Главная страница Promokods.BY представлена на рисунке 1.7.

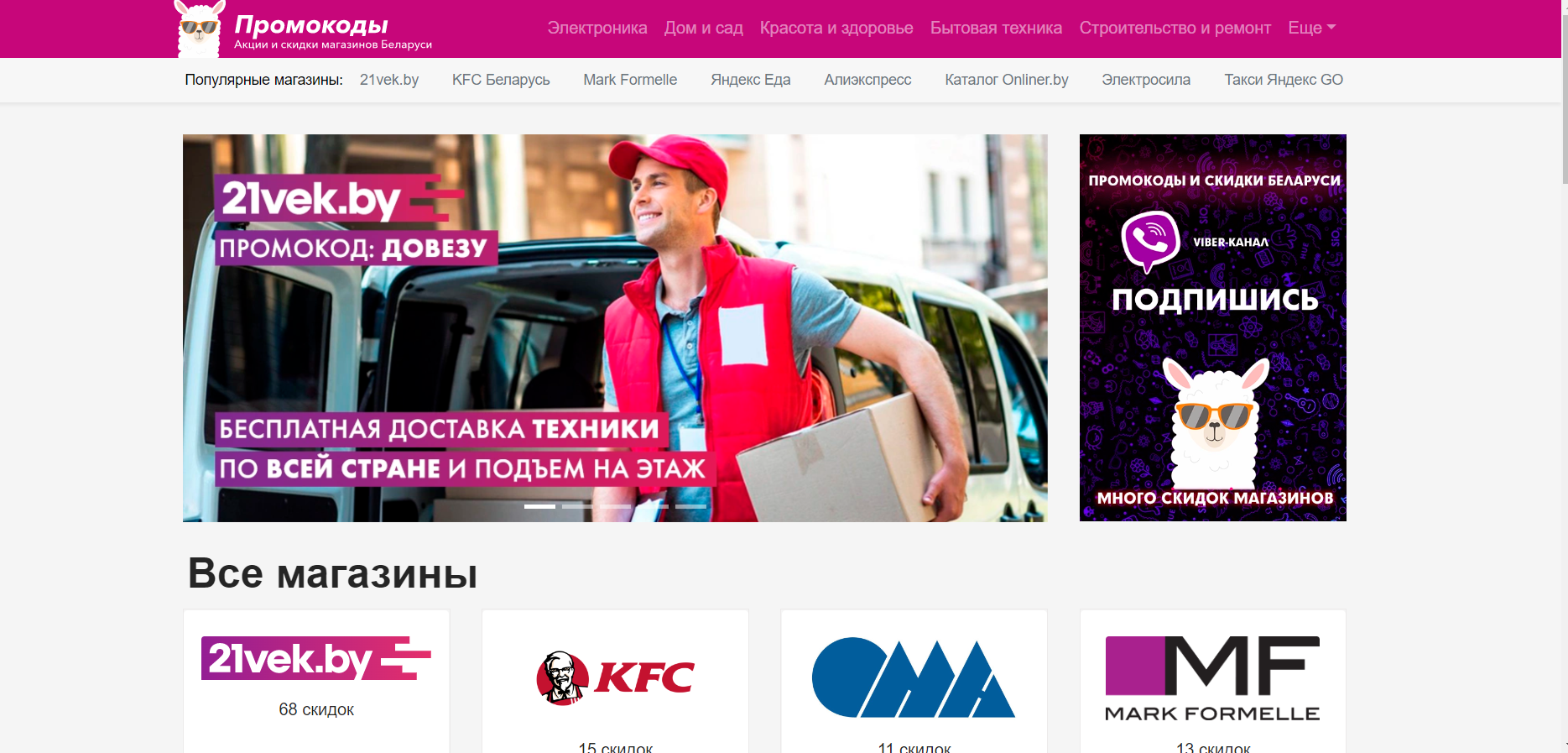


Рисунок 1.7 – Главная страница Promokods.BY

Весь сайт разделен на блоки, которые в свою очередь содержат определённое количество скидок.

Пример такого блока представлен на рисунке 1.8.

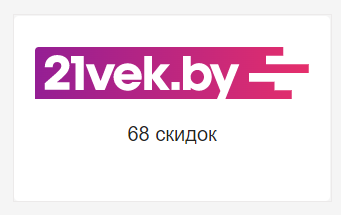


Рисунок 1.8 – Пример блока 21vek.by

Так же, как и в Slivki.BY мы можем воспользоваться промокодом для получения нужной скидки.

Список промокодов компании 21vek.by представлен на рисунке 1.9.

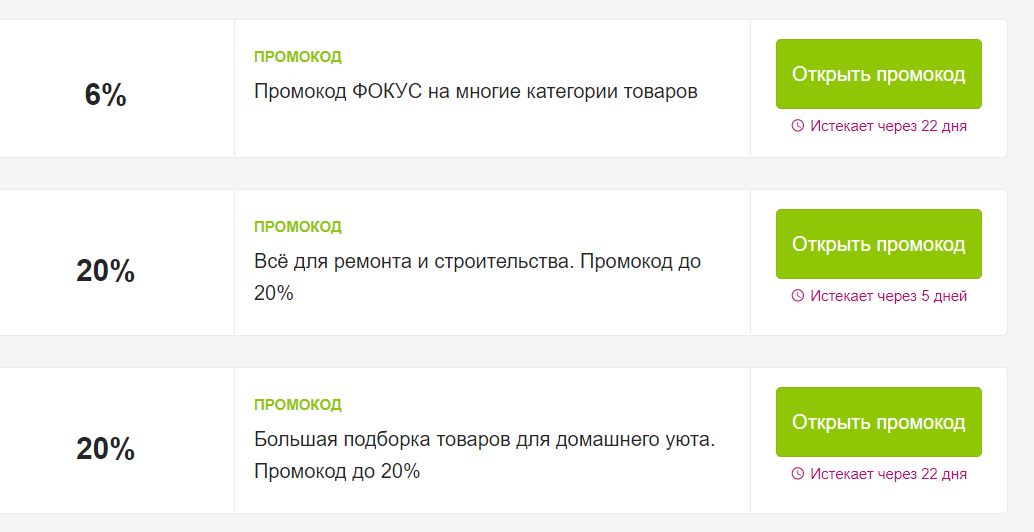


Рисунок 1.9 – Список промокодов компании 21vek.by

Перейдя по одному из промокодов, мы можем получить скидку по определённой категории товаров или услуг. При этом, для получения скидки, пользователь должен ввести промокод в специальном поле при оформлении заказа на сайте продавца. Некоторые промокоды могут быть ограничены по сроку действия, сумме покупки или другим условиям. Всю необходимую информацию о промокоде можно найти на сайте.

Пример промокода для сайта 21vek.by представлен на следующем рисунке 1.10.

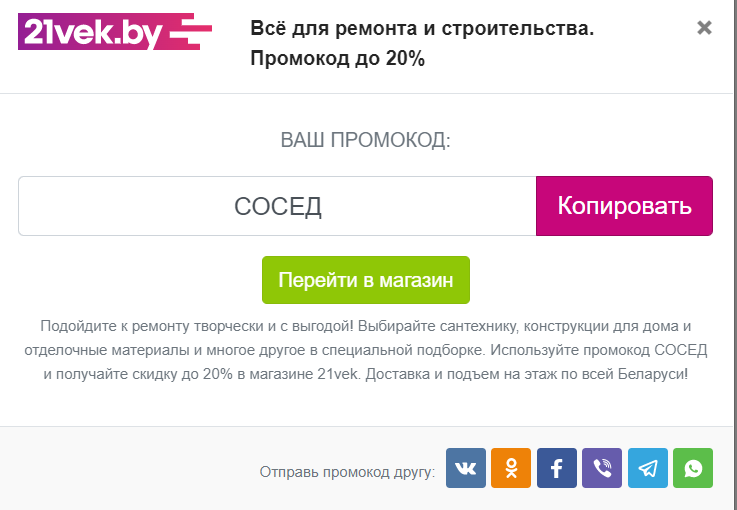


Рисунок 1.10 – Пример промокода для сайта 21vek.by

После изучения двух аналогов нашего приложения, мы можем перейти к разработке функциональных задач проекта.

* 1. Выводы по разделу

Подведем итоги по основным функциональным задачам проекта:

* обеспечивать возможность регистрации и авторизации пользователей;
* осуществлять разделение по ролям (администратор компании, пользователь);
* обеспечивать возможность добавления своих компаний;
* осуществлять возможность добавления товаров и скидок;
* обеспечивать возможность добавлять компанию в избранное;
* обеспечивать возможность активации купона для пользователя;
* обеспечивать просмотр своей статистики активаций купонов;
* обеспечивать просмотр статистики активированных купонов для компании;
* обеспечивать возможность сортировки компаний;

Следуя основной сути проекта и его функциональным задачам, можно составить цели работы:

* создание централизованной базы данных и ее наполнение;
* создание Back-end части на ASP.NET Core;
* создание Front-end части на Angular;
* связь приложения с базой данных;
* реализация алгоритма авторизации пользователей;
* настройка параметров маршрутизации страниц;
* разграничение прав доступа;
* создание простого и понятного интерфейса;

1. Проектирование
   1. Выбор инструментов разработки

В качестве используемой базы данных будет выбрана популярная СУБД SQLite. Она является полностью бесплатной, удобной и практичной в использовании.

Следующим этапом для выбора разработки веб-приложения будет платформа для написания Back-end части.

C# (произносится си шарп) — объектно-ориентированный язык программирования. Разработан в 1998—2001 годах группой инженеров компании Microsoft под руководством Андерса Хейлсберга и Скотта Вильтаумота как язык разработки приложений для платформы Microsoft .NET Framework. Впоследствии был стандартизирован как ECMA-334 и ISO/IEC 23270[1]. C# относится к семье языков с C-подобным синтаксисом, из них его синтаксис наиболее близок к C++ и Java. Язык имеет статическую типизацию, поддерживает полиморфизм, перегрузку операторов (в том числе операторов явного и неявного приведения типа), делегаты, атрибуты, события, переменные, свойства, обобщённые типы и методы, итераторы, анонимные функции с поддержкой замыканий, LINQ, исключения, комментарии в формате XML.

Переняв многое от своих предшественников — языков C++, Delphi, Модула, Smalltalk и, в особенности, 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).

С# разрабатывался как язык программирования прикладного уровня для CLR и, как таковой, зависит, прежде всего, от возможностей самой CLR. Это касается, прежде всего, системы типов С#, которая отражает BCL. Присутствие или отсутствие тех или иных выразительных особенностей языка диктуется тем, может ли конкретная языковая особенность быть транслирована в соответствующие конструкции CLR. Так, с развитием CLR от версии 1.1 к 2.0 значительно обогатился и сам C#; подобного взаимодействия следует ожидать и в дальнейшем (однако, эта закономерность была нарушена с выходом C# 3.0, представляющего собой расширения языка, не опирающиеся на расширения платформы .NET). CLR предоставляет С#, как и всем другим .NET-ориентированным языкам, многие возможности, которых лишены «классические» языки программирования. Например, сборка мусора не реализована в самом C#, а производится CLR для программ, написанных на C# точно так же, как это делается для программ на VB.NET, J# и др.

Существует несколько реализаций C#: Реализация C# в виде компилятора csc.exe включена в состав .NET Framework (включая .NET Micro Framework, .NET Compact Framework и его реализации под Silverlight и Windows Phone 7). В составе проекта Rotor (Shared Source Common Language Infrastructure) компании Microsoft. Проект Mono включает в себя реализацию C# с открытым исходным кодом. · Проект DotGNU также включает компилятор C# с открытым кодом. DotNetAnywhere — ориентированная на встраиваемые системы реализация CLR, поддерживает практически всю спецификацию C# 2.0.

Когда говорят C#, нередко имеют в виду технологии платформы .NET (Windows Forms, WPF, ASP.NET, Xamarin). И, наоборот, когда говорят .NET, нередко имеют в виду C#. Однако, хотя эти понятия связаны, отождествлять их неверно. Язык C# был создан специально для работы с фреймворком .NET, однако само понятие .NET несколько шире.

Как-то Билл Гейтс сказал, что платформа .NET — это лучшее, что создала компания Microsoft. Возможно, он был прав. Фреймворк .NET представляет мощную платформу для создания приложений. Можно выделить следующие ее основные черты:

Поддержка нескольких языков. Основой платформы является общеязыковая среда исполнения Common Language Runtime (CLR), благодаря чему .NET поддерживает несколько языков: наряду с C# это также VB.NET, C++, F#, а также различные диалекты других языков, привязанные к .NET, например, Delphi.NET. При компиляции код на любом из этих языков компилируется в сборку на общем языке CIL (Common Intermediate Language) - своего рода ассемблер платформы .NET. Поэтому при определенных условиях мы можем сделать отдельные модули одного приложения на отдельных языках.

Кроссплатформенность. .NET является переносимой платформой (с некоторыми ограничениями). Например, последняя версия платформы на данный момент - .NET 6 поддерживается на большинстве современных ОС Windows, MacOS, Linux. Используя различные технологии на платформе .NET, можно разрабатывать приложения на языке C# для самых разных платформ - Windows, MacOS, Linux, Android, iOS, Tizen.

Мощная библиотека классов. .NET представляет единую для всех поддерживаемых языков библиотеку классов. И какое бы приложение мы не собирались писать на C# - текстовый редактор, чат или сложный веб-сайт - так или иначе мы задействуем библиотеку классов .NET.

Разнообразие технологий. Общеязыковая среда исполнения CLR и базовая библиотека классов являются основой для целого стека технологий, которые разработчики могут задействовать при построении тех или иных приложений. Например, для работы с базами данных в этом стеке технологий предназначена технология ADO.NET и Entity Framework Core. Для построения графических приложений с богатым насыщенным интерфейсом - технология WPF и WinUI, для создания более простых графических приложений - Windows Forms. Для разработки кроссплатформенных мобильных и десктопных приложений - Xamarin/MAUI. Для создания веб-сайтов и веб-приложений - ASP.NET и т.д.

К этому стоит добавить активной развивающийся и набирающий популярность Blazor - фреймворк, который работает поверх .NET и который позволяет создавать веб-приложения как на стороне сервера, так и на стороне клиента. А в будущем будет поддерживать создание мобильных приложений и, возможно, десктоп-приложений.

Производительность. Согласно, ряду тестов веб-приложения на .NET 6 в ряде категорий сильно опережают веб-приложения, построенные с помощью других технологий. Приложения на .NET 6 в принципе отличаются высокой производительностью.

Также еще следует отметить такую особенность языка C# и фреймворка .NET, как автоматическая сборка мусора. А это значит, что нам в большинстве случаев не придется, в отличие от С++, заботиться об освобождении памяти. Вышеупомянутая общеязыковая среда CLR сама вызовет сборщик мусора и очистит память [3].

И, последним этапом в выборе инструментов, является выбор UI-фреймворка для создания Front-end части. На данный момент их выбор на рынке достаточно большой, можно привести такие примеры, как: Angular, Vue.JS, Ember. Однако, мой выбор был сделан в пользу Angular, поскольку он имеет ряд существенных преимуществ в сравнении с другими фреймворками.

Прежде всего он нацелен на разработку SPA-решений (Single Page Application), то есть одностраничных приложений. В этом плане Angular является наследником другого фреймворка AngularJS. В то же время Angular это не новая версия AngularJS, а принципиально новый фреймворк.

Angular предоставляет такую функциональность, как двустороннее связывание, позволяющее динамически изменять данные в одном месте интерфейса при изменении данных модели в другом, шаблоны, маршрутизация и так далее.

Одной из ключевых особенностей Angular является то, что он использует в качестве языка программирования TypeScript. Поэтому перед началом работы рекомендуется ознакомиться с основами данного языка, про которые можно прочитать здесь.

Но мы не ограничены языком TypeScript. При желании можем писать приложения на Angular с помощью таких языков как Dart или JavaScript. Однако TypeScript все-таки является основным языком для Angular.

* 1. Диаграмма вариантов использования

Диаграмма вариантов использования — это исходное концептуальное представление или концептуальная модель системы в процессе ее проектирования и разработки. Диаграмма состоит из актеров, вариантов использования и отношений между ними. При построении диаграммы могут использоваться также общие элементы нотации: примечания и механизмы расширения.

Веб-приложение включает в себя администратора и пользователя.

Пользователи должны обязательно быть авторизированными для активации купона. Для этого нужно пройти самую обычную регистрацию.

Авторизированный пользователь имеет доступ ко всем базовым функциями приложения: просмотр компаний, активация купона, добавление компании в избранное, просмотр товаров компании, а также поиск нужного товара.

Администратор владеет функциями просмотра компаний, создание своей компании, добавление и удаление своих товаров, просмотр статистики своей компании.

На рисунке 2.1 изображена диаграмма вариантов использования, на которой показаны варианты использования веб-приложения.

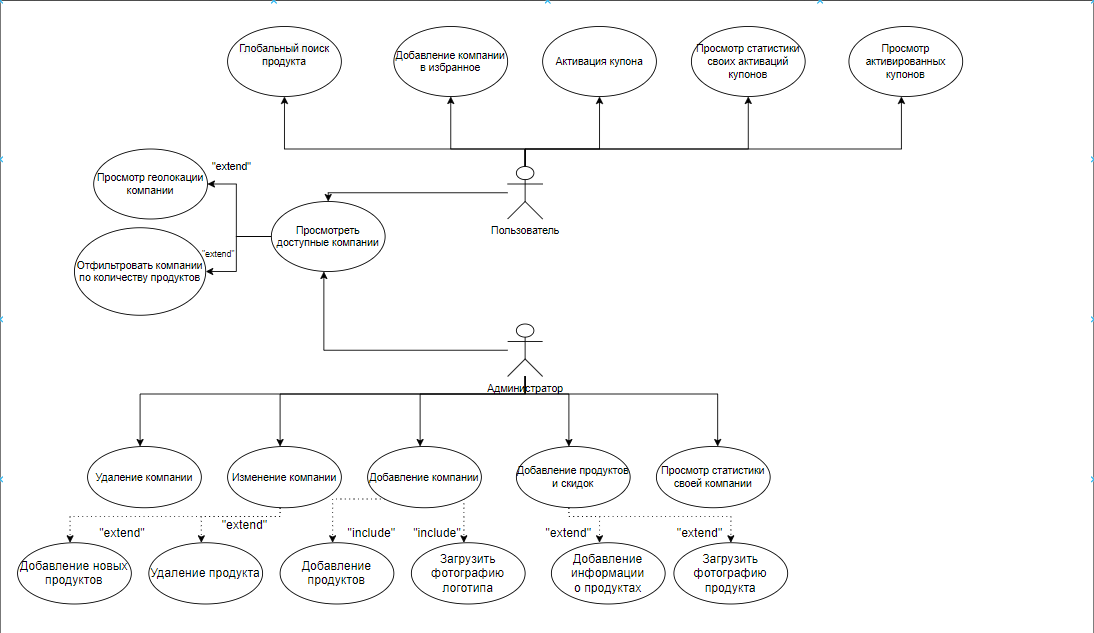


Рисунок 2.1 – Диаграмма варианта использования

На примере алгоритма создания компании построим блок схему. Блок схема представлена на рисунке 2.2.

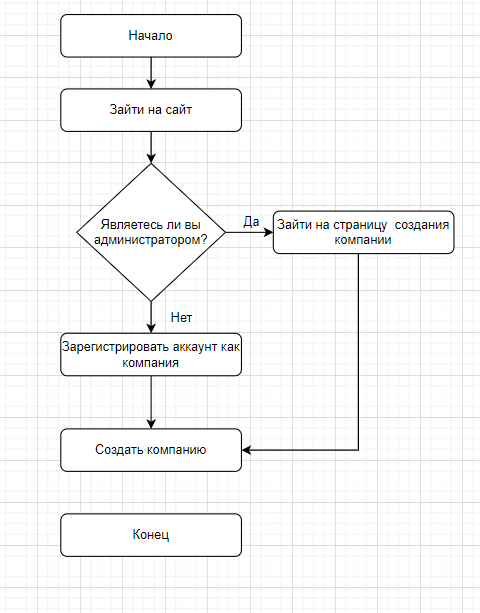


Рисунок 2.2 – Блок схема алгоритма создания компании

После построения и изучения диаграммы вариантов использования и блок схемы перейдем к изучению архитектуры приложения.

* 1. Архитектура

Общая архитектура приложения представлена на рисунке 2.3.

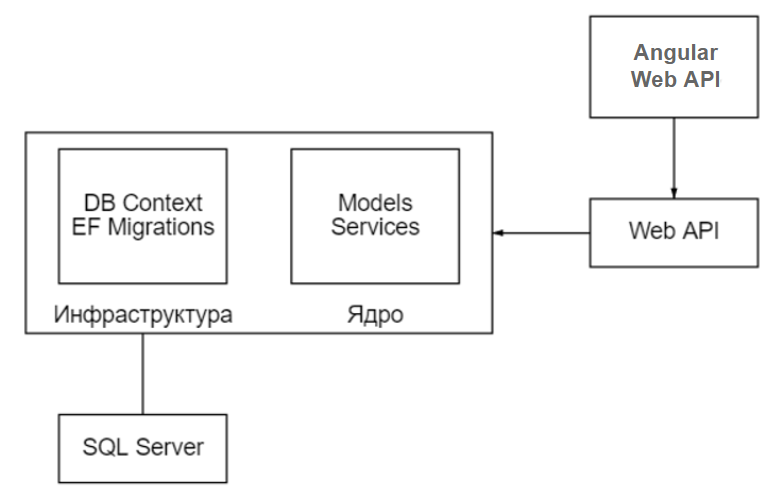


Рисунок 2.3 – Общая архитектура приложения

Использование приложения начинается с пользовательского интерфейса, который реализован при помощи фреймворка Angular. Пользователь выполняет действие, посылает запрос по адресу развернутого на сервере IIS веб-API приложения. В свою очередь управление переходит к сервисам ASP.NET Core и модели. Если запрос требует работы с данными, используется Entity Framework и контекст базы данных. SQL Server выдает необходимые данные. Entity Framework передает эти данные веб-API, который формирует ответ пользователю.

На рисунке 2.4 изображена клиент-серверная архитектура приложения.

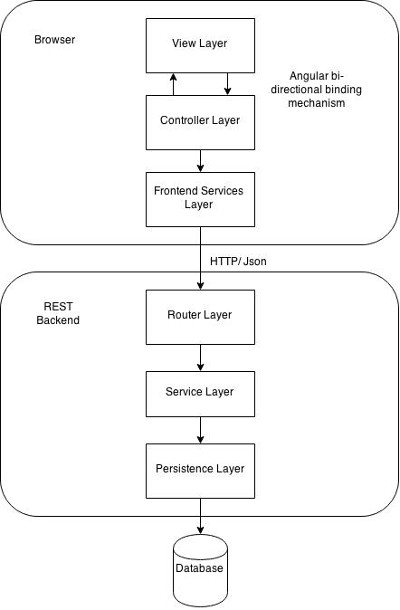


Рисунок 2.4 – Клиент-серверная архитектура

Важно отметить, что подобная архитектура является классическим примером «N-layer» архитектуры, где каждый слой зависит только от предыдущего компонента. Вообще существует множество различных видов и типов архитектур, которые успешно применяются. Одной их наиболее используемых является классическая трехуровневая система, которая подразумевает разделение приложения на три уровня. Тут сразу надо сказать, что многоуровневой архитектурой часто обозначают два не совсем связанных понятия: n-layer и n-tier. И layer, и tier, как правило, обозначаются словом "уровень", иногда по отношению к "layer" еще употребляется слово "слой". Однако в обоих случаях уровни будут разного порядка. Также взаимодействие происходит последовательно. Так, к примеру, последний компонент не способен взаимодействовать с первым. Всего выделяют три абстрактных уровня: уровень представления, необходимый для работы пользователя, уровень бизнес-логики, где осуществляется основная обработка данных, и уровень доступа к данным, необходимый для их хранения и использования. При этом надо отметить, что крайние уровни не могут взаимодействовать между собой, то есть уровень представления (применительно к ASP.NET MVC, контроллеры) не могут напрямую обращаться к базе данных и даже к уровню доступа к данным, а только через уровень бизнес-логики.

* 1. Разработка модели базы данных

Как отмечалось в предыдущем разделе, одним из ключевых моментов при проектировании и создании базы данных является грамотный анализ предметной области приложения. Как следствие – составление такой модели данных, которая будет правильно отражать то, как с этими с данными в общем, и этой моделью, в частности, подразумевается взаимодействовать.

Результатом корректно проведённого анализа, проектирования, и разработки, является модель, способная предоставить функционально все необходимые возможности для пользователя. В частности, для моего приложения, можно выделить набор таких операций, как:

* возможность авторизации пользователей под разными ролями;
* возможность создания компаний;
* возможность создания товаров;
* возможность удаления товаров;
* возможность просмотра статистики;
* возможность добавления компании в избранное;
* возможность поиска нужного товара;
* возможность активации купона;
* возможность просмотра расположения компании на карте;
* возможность сортировки компаний;

На рисунке 2.5 изображена схема наиболее последней актуальной базы данных, разработанной при подготовке к созданию веб-приложения и реализацией необходимого функционала. Между сущностями, представленными таблицами, также настроены связи, отражающие их взаимосвязь.



Рисунок 2.5 – Диаграмма базы данных

В базе данных реализовано 5 основных таблиц. Теперь рассмотрим их поподробнее.

Таблица «AspNetUsers» – содержит данные о каждом зарегистрированном в системе пользователе, а именно почту, захэшированный пароль, имя, фамилию, а также роль. Типичное количество пользователей в реальной системе может достигать до миллиона человек.

Таблица «Companies» – таблица, содержащая информацию о компаниях, их адрес, описание, идентификатор компании.

Таблица «Products» – таблица, характеризующая товары в нашем приложении. Именно в этой таблице содержится вся доступная для просмотра информация о товаре.

Таблица «Favorites» – таблица, в которой содержатся все понравившиеся компании для пользователя. В ней хранится информация как о компании, также и о пользователе.

Таблица «OrderedProducts» – таблица, в которой содержатся все активированные купоны для пользователя. В ней хранится информация как о товаре, также и о пользователе.

Теперь перейдем к рассмотрению полей таблиц базы данных.

Таблица «AspNetUsers»:

* Id (uniqueidentifier) (PK, not null) – уникальный идентификатор пользователя;
* IsAdmin(bool) (not null) - роль пользователя.
* Email (nvarchar) – логин пользователя;
* Password (nvarchar) – хэш пароля;
* Name (nvarchar) – имя пользователя;
* Surname (nvarchar) – фамилия пользователя;

Таблица «Companies»:

* Id (uniqueidentifier) (PK, not null) – уникальный идентификатор компании;
* Name (nvarchar) – название компании;
* Addres (nvarchar) – адрес компании;
* ImageSource(nvarchar) – ссылка на изображение;
* Phone(nvarchar) – номер телефона компании;
* Description(nvarchar) – описание компании;
* CreatorId(FK, uniqueidentifier) - уникальный идентификатор пользователя.

Таблица «Products»:

* Id (uniqueidentifier) (PK, not null) – уникальный идентификатор товара;
* Name (nvarchar) – название товара;
* Price(decimal) – цена товара;
* SoldCounter(int) – количество активаций данного товара;
* ImageSource(nvarchar) – ссылка на изображение;
* Description (nvarchar) – описание товара;
* CopmpanyId (FK, not null) – уникальный идентификатор компании, которому принадлежит данный товар.

Таблица «Favorites»:

* Id (uniqueidentifier) (PK, not null) – уникальный идентификатор избранной компании;
* UserId (FK, not null) – уникальный идентификатор пользователя, который добавил компанию в избранные.
* CompanyId (FK, not null) – уникальный идентификатор компании, которую добавили в избранное.

Таблица «OrderedProducts»:

* Id (uniqueidentifier) (PK, not null) – уникальный идентификатор выбранного товара;
* UserId (FK, not null) – уникальный идентификатор пользователя, который выбрал данный товар.
* ProductId (FK, not null) – уникальный идентификатор товара, который был выбран пользователем.
  1. Выводы по разделу

В данном разделе мы описали технологии, с помощью которых будет реализовано разрабатываемое программное средство, а также описали архитектуру взаимодействия наших frontend и backend частей разрабатываемого приложения.

# Заключение

В ходе выполнения работы были проанализированы аналоги веб-приложения по предоставлению рекламы компаниям и скидкам для пользователям, выявлены их положительные и отрицательные стороны. В свою очередь, результаты анализа были учтены при дальнейшем проектировании дизайна и функционала приложения.

Была разработана реляционная база данных, использующая язык SQL и СУБД SQLite, имеющая в своем наборе 5 обязательных таблиц. В проекте использовался подход Code First Entity Framework, поскольку в нашем случае он подходит больше всего по принципу использованию и архитектуре.

Было разработано и Front-end приложение на современном фреймворке Angular, позволяющее создавать гибкую и удобную верстку, а также легко взаимодействовать с запросами. В нашем случае в проекте активно использовались http-запросы, написанные на языке Type Script.

Говоря о Back-end части, проект четко разделен на библиотеки и backend модули. Это позволит в дальнейшем расширять приложение без каких-либо проблем, поскольку мы с лёгкостью сможем подключать нужный нам компонент.

Также была осуществлена разработка авторизации пользователей по их ролям, а именно в системе имеется на данный момент две роли – администратор и пользователь.

Отметим, что в проекте используется хранение сессий, а также шифрование и хеширование паролей.

Благодаря проведенной работе, был создан проект, который может быть очень удобен и полезен для малых и не только компаний в качестве площадки по предоставлению рекламы, а также обычным пользователям, которые хотят сэкономить на доступных товарах.

# Список использованной литературы

1. «Slivki.BY» [Электронный ресурс]. – Режим доступа: https://www.slivki.by/. – Дата доступа: 21.11.2022.
2. «Promokods.BY» [Электронный ресурс]. – Режим доступа: https://promokods.by/. – Дата доступа: 21.11.2022.
3. Professorweb [Электронный ресурс]. – Электронные данные. – Режим доступа: https://professorweb.ru/my/ASP\_NET/sites/level3/3\_2.php. – Дата доступа: 01.12.2022
4. Code.tutsplus [Электронный ресурс]. – Электронные данные. – Режим доступа: <https://code.tutsplus.com/ru/tutorials/creating-your-first-angular-app-basics--cms-30092>. – Дата доступа: 07.12.2022