### МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕУЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА»**

**(БГТУ им. В.Г. Шухова)**

### Кафедра программного обеспечения вычислительной техники и автоматизированных систем

**Курсовая работа**

По дисциплине: Основы программирования

Тема: Создание прототипа интернет-магазина

Автор работы \_\_\_\_\_\_ Маркевич Александр Александрович

(подпись)

Автор работы \_\_\_\_\_\_ Максименко Александр Михайлович

(подпись)

Руководитель проекта \_\_\_\_\_\_ Черников Сергей Викторович

(подпись)

Оценка \_\_\_\_\_\_

Белгород 2023

**Содержание**

1 Введение…………………………………………………1-2

1.1 Выбор подхода………………………………………...3-4

2 Алгоритмизация frontend……………………………….5-10

2.1 Тестирование frontend………………………………...11-14

3 Алгоритмизация backend……………………………….15-17

3.1 Тестирование backend………………………………...18-21

4 Заключение………………………………………………22

5 Список литературы……………………………………...23

6 Приложение с кодом…………………………………….24

**Глава 1. Введение**

Современный мир невозможно представить без Интернета. Он стал неотъемлемой частью жизни миллиардов людей по всему миру и существенно изменил способ, которым мы общаемся, работаем, учимся и развлекаемся. Интернет является глобальной сетью, объединяющей множество компьютеров и устройств, и предоставляющей доступ к огромному объему информации, услуг и возможностей.

Он применяется в различных целях, начиная от развлечений, заканчивая ведением бизнеса. Уже не секрет, что в интернете можно купить что угодно. Зачастую нам не сильно хочется идти в магазин и покупать, к примеру, одежду которой там может и не быть. Для этого и были созданы интернет-магазины — формы электронной коммерции, которые позволяют потребителям напрямую покупать товары или услуги у продавца.

Создание интернет-магазина стало популярным направлением в сфере разработки веб-приложений. В современном мире все больше людей предпочитают покупать товары онлайн, и интернет-магазины становятся неотъемлемой частью коммерческой деятельности.

Цель данной курсовой работы заключается в разработке поиска и сортировки товаров интернет-магазина с использованием ASP.NET Core и Entity Framework. ASP.NET Core — это прикладной фреймворк, который позволяет создавать мощные и масштабируемые веб-приложения. Entity Framework — это инструмент для работы с базой данных, который упрощает взаимодействие с данными.

Объект исследования является процесс разработки интернет-магазина.

Предмет исследования – языки программирования C# и Typescript, язык разметки и стилей — HTML5 и SCSS.

Цель – программно реализовать алгоритмы сортировки и поиска, а также разработать прототип интернет-магазина.

Для достижения поставленной цели необходимо решить следующие задачи:

* Применить на практике необходимый стэк технологий.
* Подготовка (подключение Nuget пакетов, необходимых сервисов, backend с frontend и т. д.).
* Разработать алгоритмы и реализовать поиск и сортировку.

**Глава 1.1 Выбор подхода**

Создание сайтов является интересным и актуальным занятием, в нем решает не только алгоритмическая составляющая, но и творческая. Воплощение интернет-магазина позволит проработать все навыки и шаблоны, так необходимые разработчикам.

Выбор пал на разработку Web-приложения через фреймворки Asp.Net Core (Web api) и Entity Framework Core, Angular, Базу данных PostgreSQL:

Entity Framework Core:

Фреймворк, который предоставляет удобный и простой способ взаимодействия с базой данных. EF Core позволяет разрабатывать приложения без написания прямых запросов к базе данных, а вместо этого основываться на объектно-ориентированной модели данных.

Asp.Net Core:

Кроссплатформенный фреймворк для разработки веб-приложений на языке C#. Он обладает высокой производительностью, долгой поддержкой, масштабируемостью и поддерживает различные платформы, такие как Windows, macOS и Linux. ASP.NET Core также предоставляет множество готовых компонентов и модулей для разработки веб-приложений.

**Связка Asp.Net Core и EF Core является одной из самых надежных связок для долгосрочной поддержки Web-приложения.**

Angular:

Фреймворк, который остается основным выбором для компании-разработчика программного обеспечения, поскольку он избавляет от необходимости создавать ненужный код. Он использует в качестве языка программирования TypeScript, что также позволяет повысить читаемость кода и совершать меньше ошибок во время разработки.

PostgreSQL:

База данных, которая поддерживает JSON и другие функции NoSQL, надежность и целостность данных. Открытое ПО.

**Стек и версии используемых технологий: C# -** версия 10.0**, .Net Core 7(Web api), TypeScript –** версия5.0, **Scss -** версия1.69.4, **HTML5**, **Angular –** версия 7, **Entity Framework Core –** версия 7, **Swagger –** версия 3.0.3, **PostgreSQL ~15.**

**Глав 2 Алгоритмизация frontend**

В данной главе будет описываться работоспособность сортировок на клиентской части приложения.

1. Сортировка цены по возрастанию:

**Описание сортировки:**

Алгоритм сортировки цены по возрастанию заключается в том, что на вход приходят не отсортированные данные, находящиеся в переменной products. Затем с помощью встроенной функции JS производится сортировка данных вида JSON по полю price. Функция сортировки возвращает параметр вида: obj1.price-obj2.price. В конце присваиваются отсортированные данные в переменную filteredListOfProducts.

**Блок-схема сортировки:**

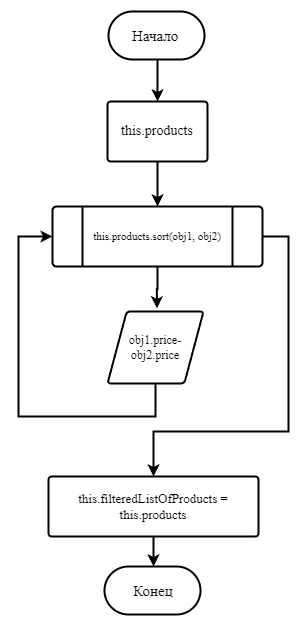


Рисунок 1. Блок-схема сортировки по возрастанию

1. Сортировка цены по убыванию:

**Описание сортировки:**

Алгоритм сортировки цены по убыванию заключается в том, что на вход приходят не отсортированные данные, находящиеся в переменной products. Затем с помощью встроенной функции JS производится сортировка данных вида JSON по полю price. Функция сортировки возвращает параметр вида: obj2.price-obj1.price. В конце присваиваются отсортированные данные в переменную filteredListOfProducts.

**Блок-схема сортировки:**

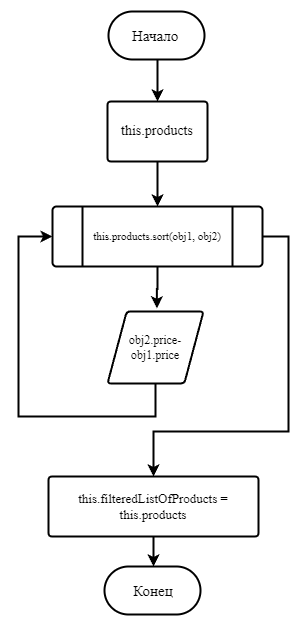


Рисунок 2. Блок-схема сортировки по убыванию

1. Сортировка слов от А до Я:

**Описание сортировки:**

Алгоритм сортировки слов от А до Я заключается в том, что на вход приходят не отсортированные данные, находящиеся в переменной products. Затем с помощью встроенной функции JS производится сортировка данных вида JSON по полю productName. Функция сортировки возвращает -1, если obj1.productName<obj2.productName. Если obj1.productName>obj2.productName, то вернет 1. Если никакое условие не сработало, то вернет 0. Встроенная функция сортировки в зависимости какие значения вернуться понимает, как отсортировать данные. В конце присваиваются отсортированные данные в переменную filteredListOfProducts.

**Блок-схема сортировки:**

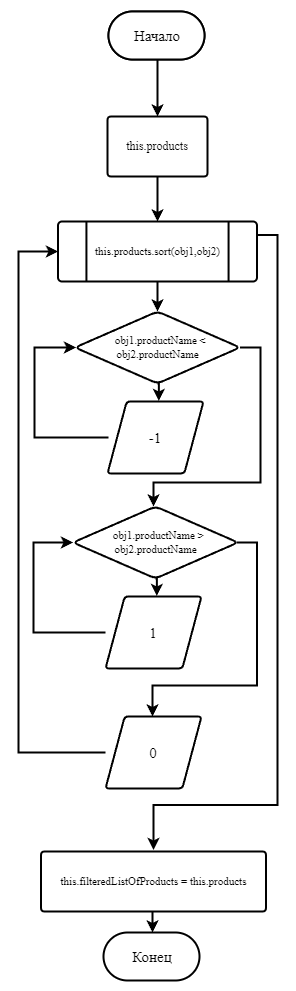


Рисунок 3. Блок-схема сортировки слов от А до Я

1. Сортировка цены в диапазоне:

**Описание сортировки:**

Алгоритм сортировки цены в диапазоне заключается в том, что на вход приходят не отсортированные данные, находящиеся в переменной products. Затем с помощью метода map перебираем каждый элемент (в коде он обозначен переменной p) и кладем этот элемент с параметром p.price в функцию isThePriceSuitable, а также еще два параметра fromWhatPrice, upToWhatPrice. Они обозначают диапазон цены, который мы хотим получить. Далее функция isThePriceSuitable возвращает 1, если price>= fromWhatPrice && price <= upToWhatPrice иначе 0. Если получили значение 1, то закидываем элемент в массив \_filteredListOfProducts иначе пропекаем элемент. Для того чтобы данные смотрелись привычнее была произведена сортировка над данными по возрастанию, которая описана выше. В конце присваиваются отсортированные данные в переменную filteredListOfProducts.

**Блок-схема сортировки:**

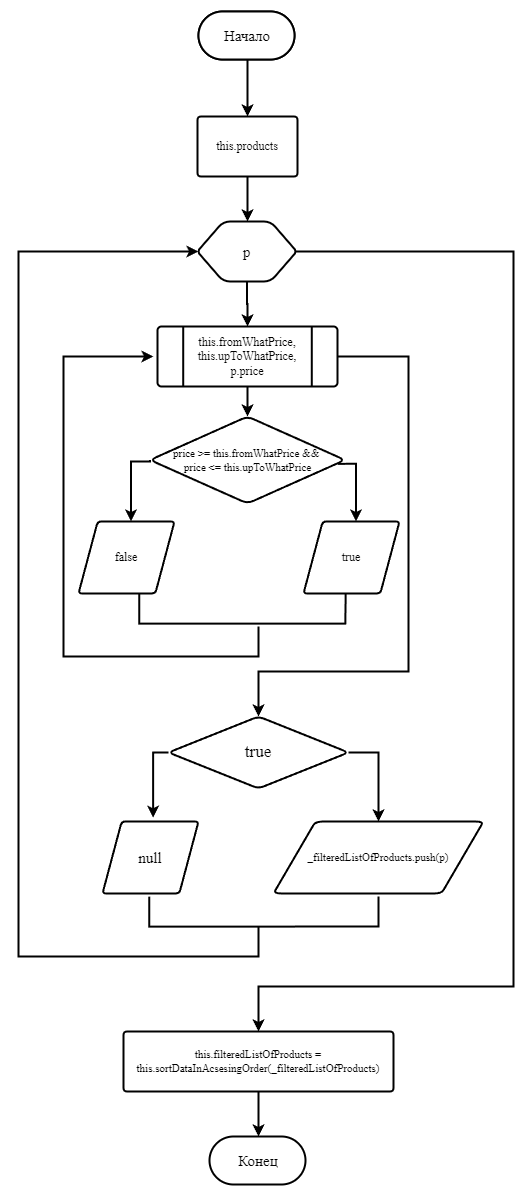


Рисунок 4. Блок-схема сортировки цены в диапазоне

**Глава 2.1 Тестирование frontend**

В данной главе будут проведены тесты на клиентской стороне приложения.

Изначальный экран приложения пользователя выглядит следующим образом. На нем отображаются данные в количестве 20 штук, которые приходят с серверной части приложения. Также имеется “шапка” сайта, в которой находится поле для поиска товаров и кнопка поиск для отправки запроса на сервер.

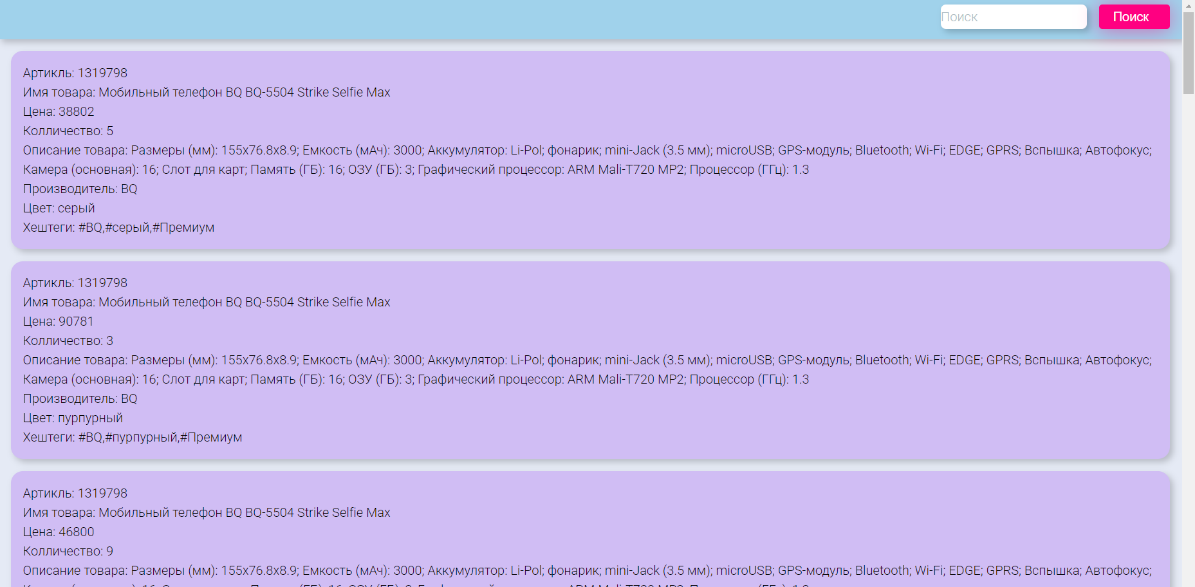


Рисунок 5. Начальный экран приложения

После поиска товара будет отображено дополнительное меню благодаря, которому можно воспользоваться различными сортировками.



Рисунок 6. Экран приложения после поиска товар

После использование сортировки по возрастанию цены данные сортируются и затем через некоторое время отобразятся на экране web-приложения. На изображении, приведенном ниже, можно заметить, что цена товаров идет по возрастанию.



Рисунок 7. Экран приложения после сортировки по возрастанию

После использование сортировки по убыванию цены данные сортируются и затем через некоторое время отобразятся на экране web-приложения. На изображении, приведенном ниже, можно заметить, что цена товаров идет по убыванию.



Рисунок 8. Экран приложения после сортировки по убыванию

После использование сортировки диапазона цены данные сортируются и затем через некоторое время отобразятся на экране web-приложения. На изображении, приведенном ниже, можно заметить, что цена товаров идет от 30000 до 50000.

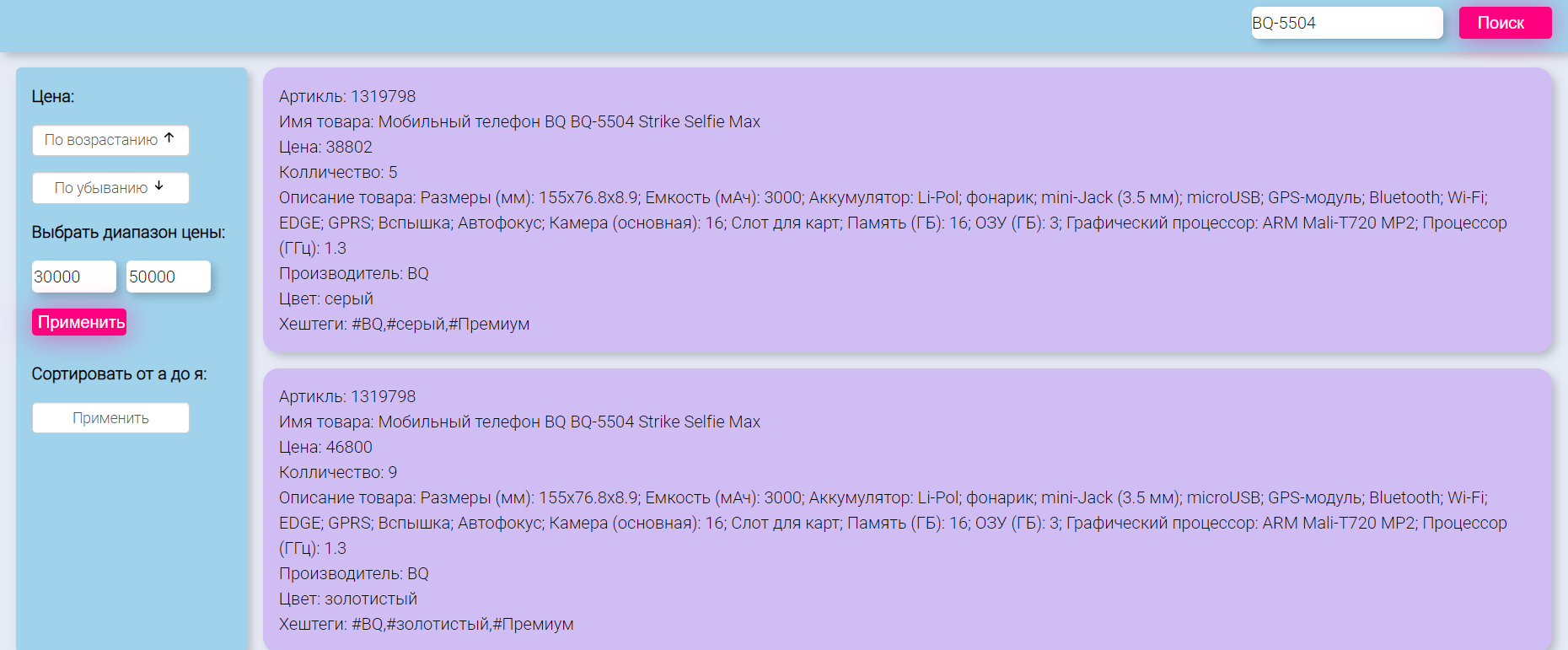


Рисунок 9. Экран приложения после сортировки по диапазону

После использование сортировки от a до я данные сортируются и затем через некоторое время отобразятся на экране web-приложения. На изображении, приведенном ниже, можно заметить, что изначально идут такие марки телефонов, как: Nokia, fly. После сортировки на экран стали отображаться марки телефонов Ark

Было:

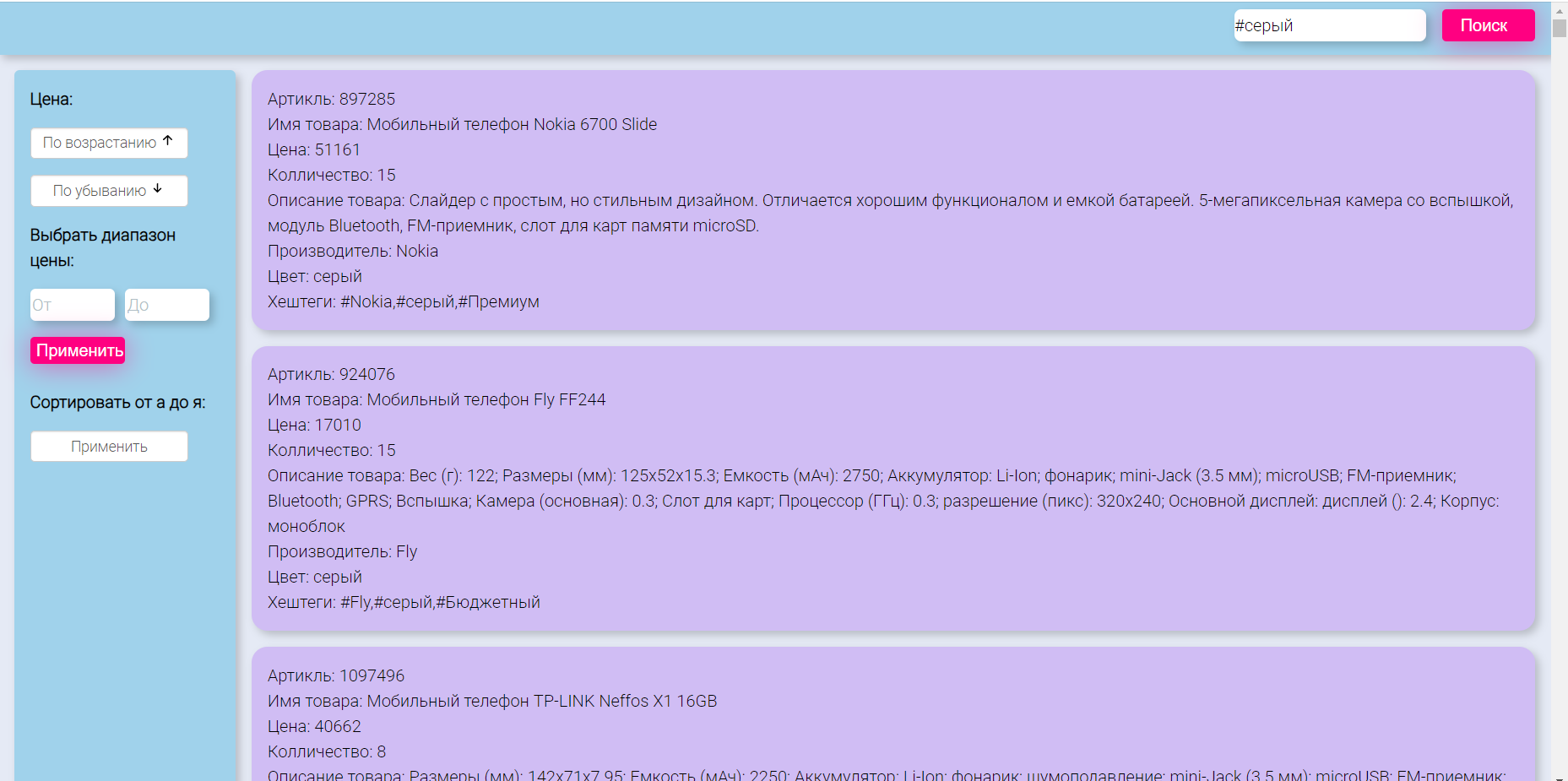


Рисунок 10. Экран приложения до сортировки от а до я

Стало:



Рисунок 11. Экран приложения после сортировки от а до я

Дополнительно. Пример ввода в поле более сложный запрос для сервера, можно заметить, что данные, приходящие с сервера верны.



Рисунок 12. Экран приложения после поиска

**Глава 3 Алгоритмизация backend**

В данной главе будет описываться с помощью блок-схем реализованные алгоритмы на серверной стороне приложения.

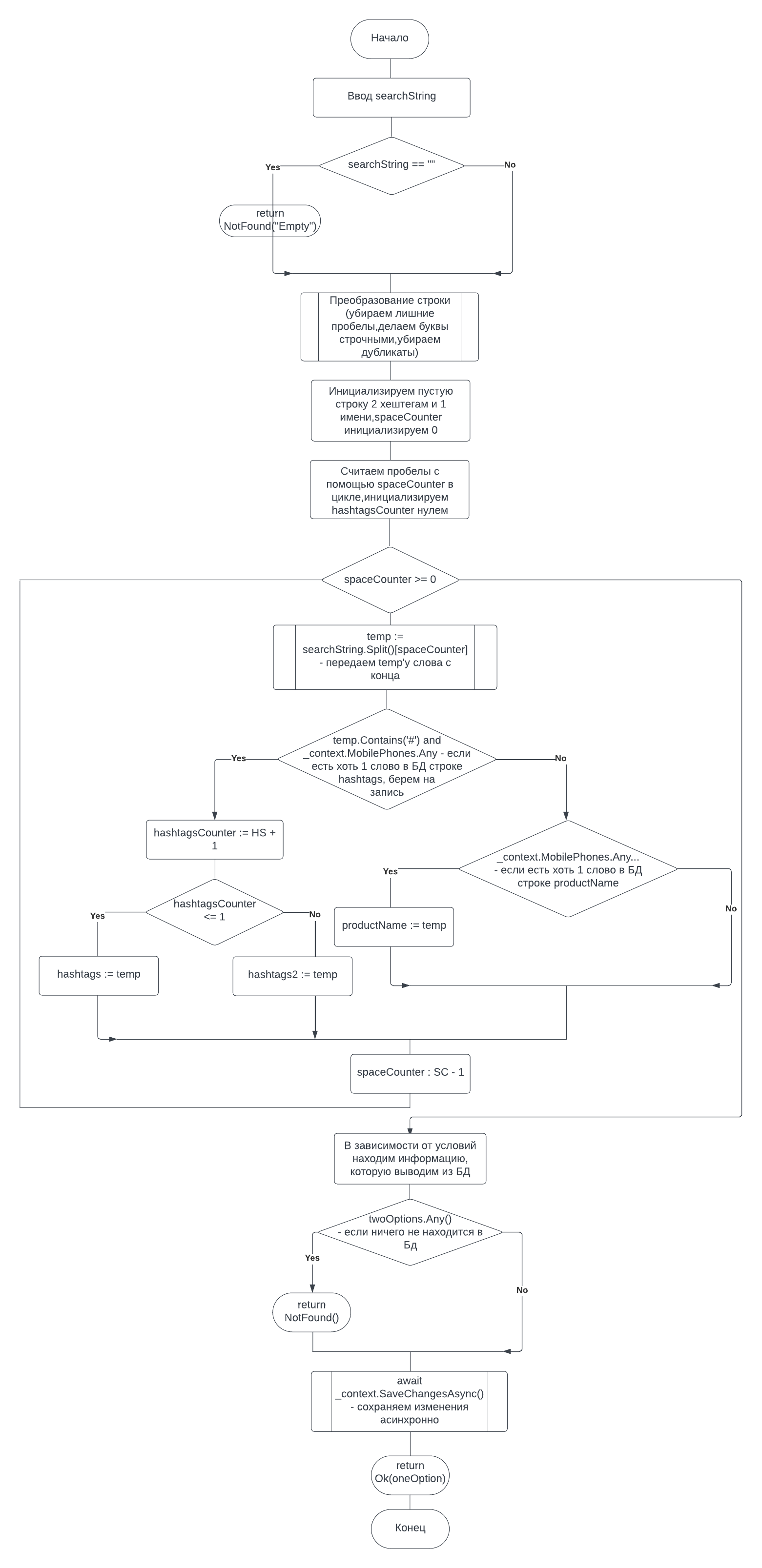


Рис. 13. Блок-схема алгоритма поиска

Описание алгоритма поиска:

Алгоритм поиска заключается в том, что при вводе нескольких слов, слова с конца переменной searchString вписываются в переменные hashtags, hashtags2, productName, если те находятся в БД, то проходятся по условиям и выводятся, но если с переменной searchString ничего не передалось в данные переменные, то пытаемся найти слово в БД, если слова нет или оно введено неверно, то выводим NotFound.

**Как выполняется алгоритм на примере:**

1. Ввод хештега #бЕлЫй
2. Преобразуем в нормальный вид (преобразуем в маленькие буквы, убираем лишние пробелы, дубликаты)
3. Ищем слово в Базе данных
4. Если нашли, то выводим **все подходящие** под хештег данные
5. Если не нашли, то ищем по другим словам, если другие слова не были найдены, то выводим **NotFound**

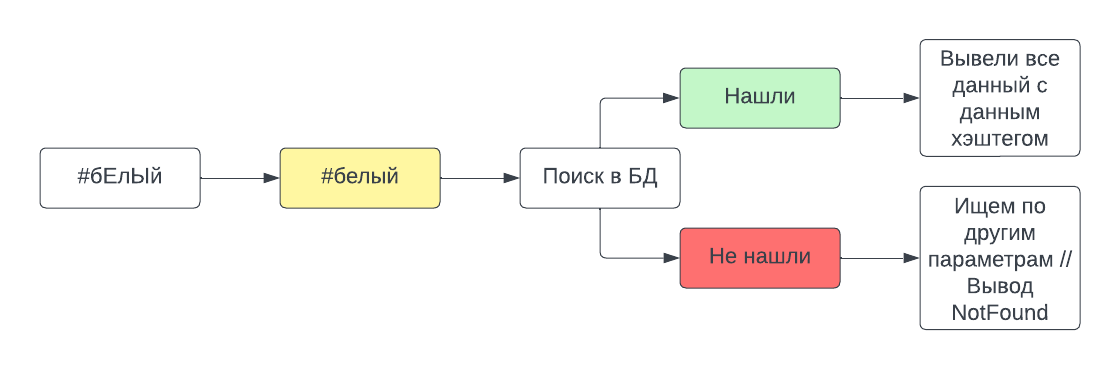


Рис. 14. Пример работы алгоритма

**Сложность данного алгоритма:**

**Худшая и Средняя – O(n) -** так как нет вложенных циклов**, Лучшая – O(1) –** так как присутствуют условия пустой строки или строки которой нет.

**Глава 3.1 Тестирование backend**

В данной главе будут проведены тесты на серверной стороне приложения. Тесты будут производиться для проверки алгоритма поиска.

Тесты имеют один и тот же алгоритм поиска:

Вводимые слова вписываются в основную строку searchString, далее проходятся по алгоритму, где идет поиск похожих слов в БД и ввод в переменные hashtags, hashtags2 и productName, если вводимые слова схожи с теми, которые находятя В БД, после этого слова проходят по условиям и выводится конечный результат.

Описания теста с 2 хештегами и 1 именем (неполные):

На вход подаются 2 хештега: #бюд #че и 1 имя: red.

Данный тест должен показать, что пользователь может ввести не полностью слова (В данном случае #бюд - #бюджетный, red – redmi, #че - #черный) и все равно найдутся нужные товары по параметрам.

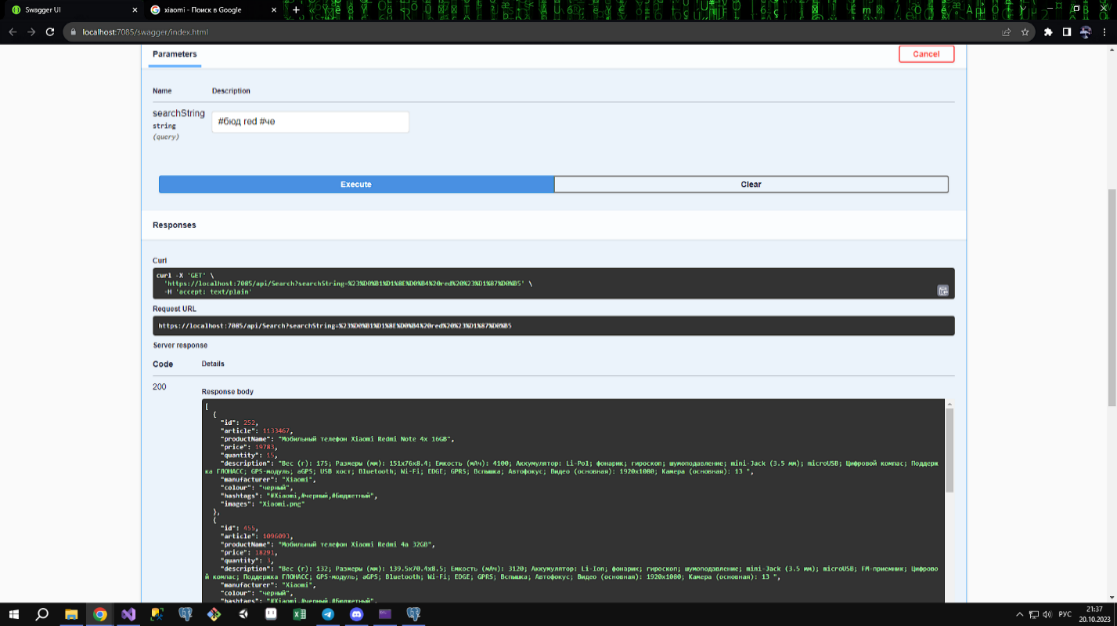


Рис. 15. Тест с 2 хештегами и 1 именем (неполные)

На вход подаются 2 хештега: #aPple #Серый

Данный тест должен показать, что ввод больших букв в разных частях слова не зависит на результат поиска, он также производится правильно, так как строка проходит через преобразования (все буквы в вводимых словах и словах в БД заменяются на строчные)

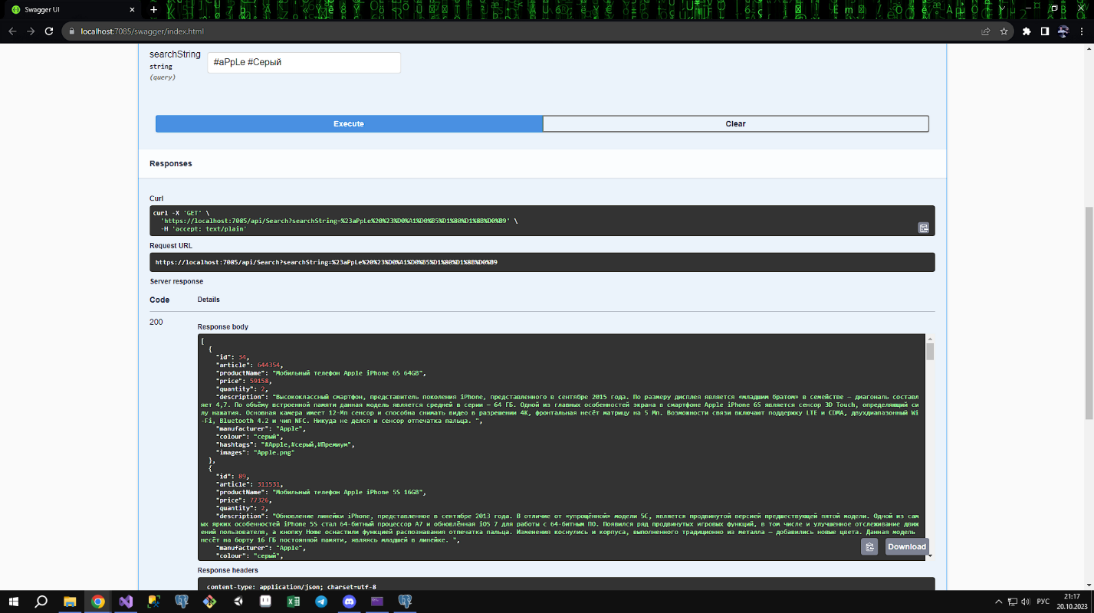


Рис. 16. Тест с 2 хештегами

На вход подается 2 хештега: #синиЙ #преМиум и 1 имя: Iphone.

Данный алгоритм должен продемонстрировать корректный поиск при вводе различных букв в разных частях слова при 3 полных опциях.

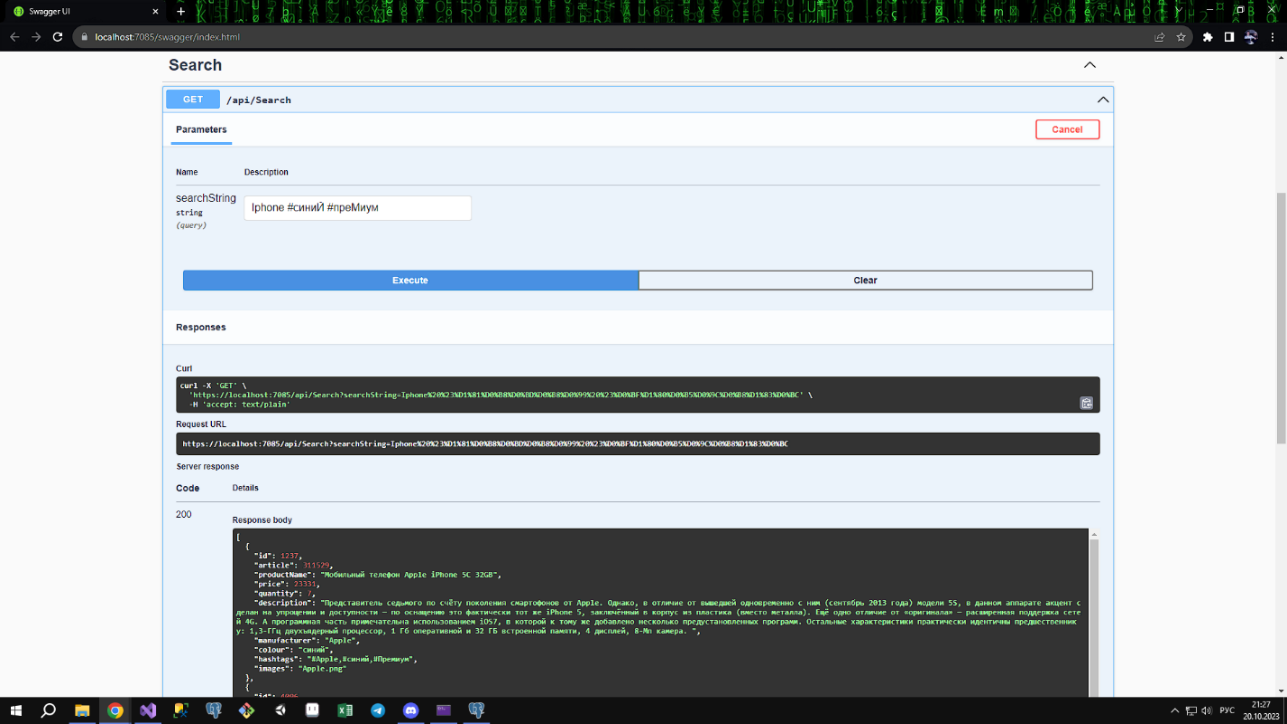


Рис. 17. Тест с 2 хэштегами и 1 именем

На вход подается 1 хештег: #bq

Данный алгоритм должен продемонстрировать поиск 1 хештега. Слово передается в переменную hashtags, проходится по условию и находится в БД.

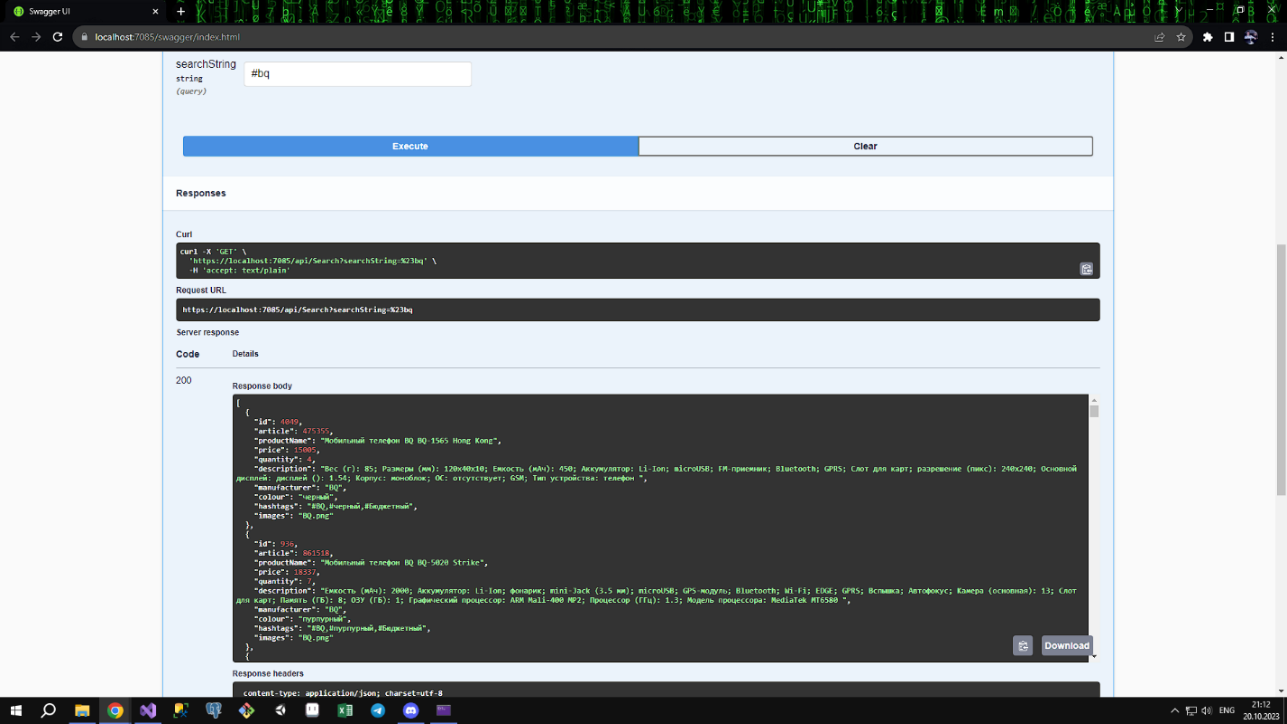


Рис. 18. Тест с 1 хэштегом

На вход подается 1 имя: SAMsung.

Данный тест показывает, что поиск может осуществляться при вводе 1 имени с большими и маленькими буквами в 1 слове. Слово передается с переменной searchString в переменную productName, проходится по условию и находится в БД.

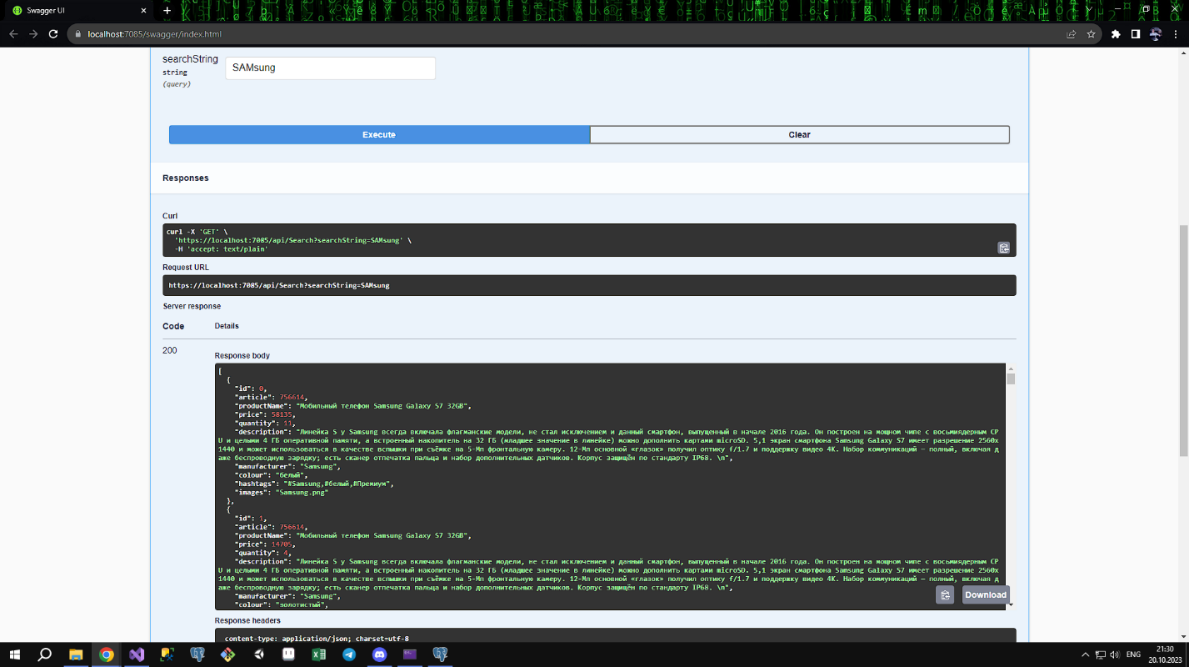


Рис. 19. Тест с 1 именем

На вход подается 1 хештег: #бюд и 1 имя: red.

В тесте показан поиск 2 не полностью введенных слов.

Слово #бюд передается в переменную hashtags, а red в переменную productName, переменные проходятся по условиям и находятся в БД.

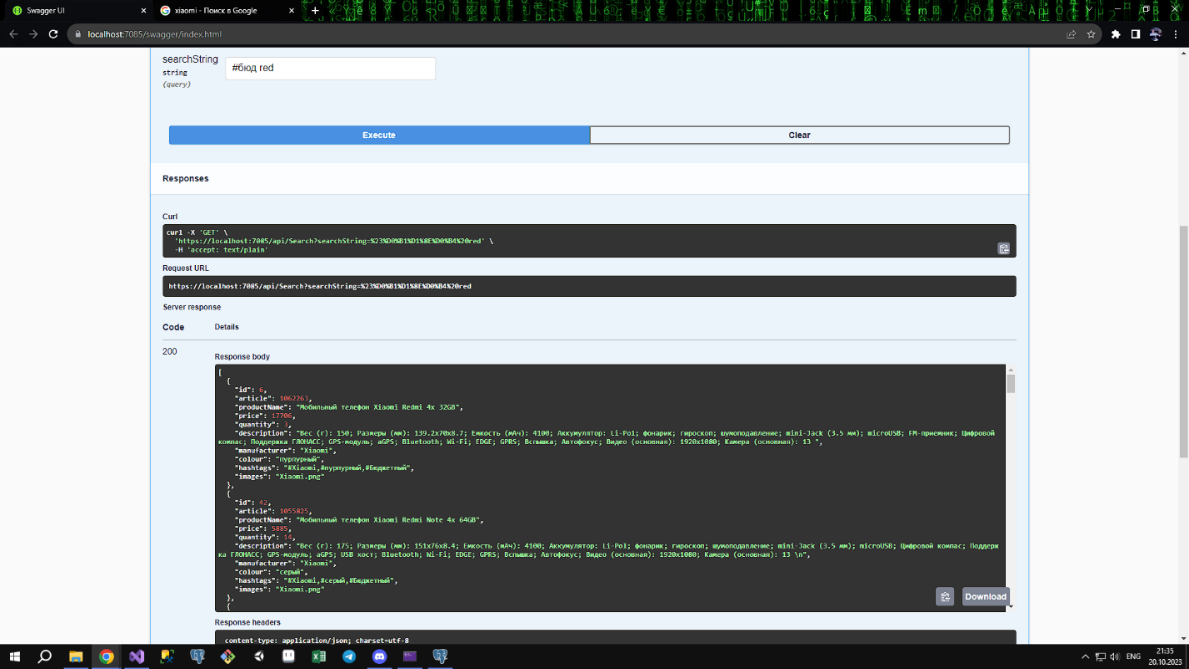


Рис. 20. Тест с 1 хештегом и 1 имене

**Глава 4 Заключение**

Программно реализованы алгоритмы сортировки и поиска, а также разработан прототип интернет-магазина. Реализация алгоритмов поиска и сортировки в интернет-магазине с использованием .NET Core, EF Core и Angular является важным шагом для оптимизации пользовательского опыта и повышения эффективности работы магазина.

**Что можно поправить**

В данном алгоритме идет уклон на запись и сохранение ограниченного числа переменных (слов) с конца строки до начала и проверка на пустую строку, также на нахождение слова в БД. Из-за этого при добавлении новых, допустим, хэштегов придется постоянно лезть в код и обновлять условия.

Поэтому лучшим решением в будущем будет – автоматизация.

Также стоит добавить картинки к товарам, стоит структурировать информацию и выводить ее в удобном виде для пользователей. В будущем можно добавить функционал, позволяющий автоматически оптимизировать алгоритмы под конкретные запросы пользователей, а также дополнительные возможности фильтрации и сортировки товаров. Также можно усовершенствовать алгоритмы для обеспечения более быстрой работы и точных результатов поиска. И конечно не стоит забывать про безопасность, она играет ключевую роль в работе веб-приложения. Добавление безопасности в веб-приложение является критически важным аспектом разработки. Это необходимо для защиты конфиденциальных данных пользователей, предотвращения утечек информации, предотвращения атак хакеров и других киберугроз.

**Глава 5 Список источников и литературы**

1. Руководство по Entity Framework Core 7 [Электронный ресурс]. Дата обращения: 08.11.2023. Режим доступа: <https://metanit.com/sharp/efcore/>
2. Документация Entity Framework Core [Электронный ресурс]. Дата обращения: 03.11.2023. Режим доступа: <https://learn.microsoft.com/ru-ru/ef/core/>
3. Документация по Angular [Электронный ресурс]. Дата обращения: 07.11.2023. Режим доступа: <https://angular.io/docs>

**Приложение с кодом:**

Ссылка на github репозиторий: [**https://github.com/Mark3vich/online\_store**](https://github.com/Mark3vich/online_store)