**Реферат**

Пояснительная записка курсового проекта содержит 36 страниц пояснительной записки, 40 иллюстраций, 5 источников литературы, 2 приложения.

MICROSOFT SQL SERVER, JAVA, SPRING FRAMEWORK, OLAP

Основная цель курсового проекта: проектирование базы данных для приложения по контролю пищевых привычек и освоение технологии OLAP.

В первой главе проводится аналитический обзор литературы по тематике курсового проекта и содержит анализ методов решения поставленных задач.

Вторая глава посвящена процессу проектирования системы.

В третьей главе приведено обоснование технических приёмов.

В четвертой главе описан процесс тестирования, позволяющий понять интерфейс программного средства и предоставить доказательства работоспособности приложения.

В заключении приведены результаты проделанной работы.

**Содержание**

[Введение 5](#_Toc89723289)

[1 Аналитический обзор литературы 6](#_Toc89723290)

[1.1 Аналитический обзор источников 6](#_Toc89723291)

[1.2 Обзор аналогов 6](#_Toc89723292)

[2 Разработка программной системы 8](#_Toc89723293)

[2.1 Разработка модели базы данных 8](#_Toc89723294)

[2.2 Хранимые процедуры 10](#_Toc89723295)

[2.2.1 Выборка данных из таблиц 10](#_Toc89723296)

[2.2.2 Выборка данных по поисковому запросу 13](#_Toc89723297)

[2.2.3 Добавление данных в таблицы 14](#_Toc89723298)

[2.2.4 Удаление данных из таблиц 14](#_Toc89723299)

[2.2.4 Изменение данных в таблицах 15](#_Toc89723300)

[2.2.5 Экспорт и импорт данных в формате JSON 15](#_Toc89723301)

[2.3 Индексы 16](#_Toc89723302)

[2.4 Многомерные структуры данных 17](#_Toc89723303)

[2.4.1 Измерения OLAP-куба 17](#_Toc89723304)

[2.4.2 OLAP-куб 19](#_Toc89723305)

[3 Обоснование технических приемов программирования 22](#_Toc89723306)

[4 Тестирование, проверка работоспособности и анализ данных 24](#_Toc89723307)

[Заключение 33](#_Toc89723308)

[Список литературы 34](#_Toc89723309)

[Приложение А 35](#_Toc89723310)

[Приложение Б 37](#_Toc89723311)

# Введение

Целью данной работы является разработка реляционной базы данных на тему «Приложение для контроля пищевых привычек». База данных должна быть составлена для возможности использования в целях хранения информации о продуктах, добавленных в коллекцию приложения пользователем, а также для хранения данных о физических параметрах тела пользователя и его личных данных. Также было необходимо разработать соответствующее приложение, для демонстрации её работы.

База данных — это организованная структура, предназначенная для хранения информации, систематизированная таким образом, чтобы эти материалы могли быть найдены и обработаны с помощью электронной вычислительной машины.

Реляционная база данных — база данных, основанная на реляционной модели данных.

В качестве СУБД для базы данных была выбрана Microsoft SQL Server, в связи с ее простотой, производительностью, широкой распространённостью и надежностью.

Необходимо создать программу для демонстрации работы базы данных. Было разработано браузерное веб-приложение Для входа в приложение необходимо пройти процедуру регистрации. После успешной регистрации пользователь имеет возможность войти в приложение под своим логином и введя пароль от своего аккаунта.

Spring Framework — универсальный фреймворк с открытым исходным кодом для Java-платформы.

# Аналитический обзор литературы

## Аналитический обзор источников

В ходе подготовки пояснительной записки была изучена специальная техническая, учебно-методическая и справочная литература, статьи и материалы, опубликованные в сети интернет.

Основная информация о работе с фреймворком Spring была получена из официальной документации

Принцип работы с SQL были получены из статьи «Подключение к базе данных». В статье было рассмотрено подключение необходимых библиотек, работа с SQL.

Концепция создания отчетов о питании пользователя была получена из статьи «Report-Driven Design». В статье была рассмотрена общая информация о создании отчёта и пример создания отчёта.

## Обзор аналогов

Для создания принципиально нового решения в виде программного продукта, используемого для контроля рациона пользователя, необходимо проанализировать уже существующее программные средства в данной сфере. Анализ достоинств и недостатков этих аналогов позволит сформировать требования к проектируемому программному средству, учитывающие опыт существующих разработок и внести в них улучшения или изменения.

В качестве исследуемых аналогов были выбраны программные продукты, которые можно отнести к группе электронных ежедневников, блокнотов, «todo-листов» и приложений, предназначенных для непосредственного для трекинга пищевых привычек, как наиболее близкие по области применения к разрабатываемому программному средству. Источником информации послужили электронные базы в сети Интернет.

В результате поиска были обнаружены следующие ресурсы:

* «Samsung Health» – это программа, которая изначально создавалась для работы с фитнес-трекерами и смарт-часами от Samsung. Но в дальнейшем в неё был внедрён целый комплекс функций для отслеживания состояния здоровья и предоставления рекомендации по соблюдению ЗОЖ. Функционал данного программного средства включает в себя:

1. Подсчитывать количество пройденных шагов. Информация берётся из носимого гаджета Samsung или из встроенного гироскопа/акселерометра. При расчёте применяются и GPS-данные, чтобы подсчёт шагов не выполнялся, например, во время поездки на авто или велосипеде.
2. Отслеживание изменений веса. Данные вводятся либо самим пользователем, либо берутся из весов с поддержкой Bluetooth или Wi-Fi-соединения.
3. Отслеживание режима сна. Также программа предоставляет статистику физической активности, при наличии фитнес-трекера или смарт-часов добавляет функцию «умного» будильника (включается, когда пользователь находится в стадии неглубокого сна).
4. Расчёт количества употреблённых и сожжённых килокалорий. Для анализа берутся данные из персонально составленного меню.
5. Контроль за уровнем сахара в крови. Данные также нужно вводить самостоятельно или использовать совместимый носимый гаджет.
6. Контроль за изменением кровяного давления. Работает только при подключении умных часов, поддерживающих измерение АД. Показатели можно вводить и самостоятельно (измеряя давление тонометром).

* Мобильное приложение «Таблица калорий» – функциональный инструмент, подходящий для отслеживания состояния здоровья. Приложение позволяет узнать не только БЖУ и калорийность продукта, но и количество соли, кальция, гликемический индекс и уровень холестерина. Что очень полезно людям, которым важно отслеживать эти показатели, особенно диабетикам. В наличии готовые тренировки, которые приложение учитывает в ежедневном плане питания. Также есть рецепты блюд для правильного питания, подсчет выпитой воды и план блюд. Физические активности делятся на категории, и «Dine4Fit» ведет статистику того или иного вида деятельности.
* Десктопное приложение «Fit Diary» – приложение, позволяющее вести подсчёт калорий. Есть возможность добавлять в базу свои продукты и упражнения, вести учёт потребления воды и изменения веса. Также программное средство предоставляет подробные графики учёта потребления БЖУ и калорий, воды, изменения веса и физических параметров.

# Разработка программной системы

При разработке курсового проекта были созданы следующие объекты:

1. Таблицы;
2. Хранимые процедуры;
3. Индексы;
4. Многомерные структуры данных.

## Разработка модели базы данных

Для данного курсового проекта была спроектирована база данных, которая получила название «DietManager». Для управления базой данных используется ПО Microsoft SQL Server 2018. Логическая схема данных представлена на рисунке 2.1.



Рисунок 2.1 – Логическая схема данных

В базе данных находится 6 таблиц. В таблице Users хранятся данные для идентификации пользователей, зарегистрированные в приложении, UsersData – личный данные зарегистрированных пользователей, UsersParams – список отчётов о физических параметрах пользователей, Reports – список отчётов употребления продуктов пользователями, Products – список продуктов, добавленных в коллекцию продуктов пользователями, и FoodCategory, которая содержит все возможные категории продуктов.

Таблица Users состоит из следующих столбцов:

* Id – тип данных int;
* IsAdmin – тип данных bit;
* UserLogin – тип данных varchar;
* UserPassword – тип данных varbinary;
* Salt – тип данных varchar.

Столбец UserLogin содержит логин пользователя, UserPassword – хешированный алгоритмом SHA1 пароль, Salt – соль, используемая при хешировании пароля, IsAdmin – логическое значение, указывающее на то, является ли пользователь администратором.

Таблица UsersData состоит из столбцов:

* Id – тип данных int;
* IdData – тип данных int;
* FullName­ – тип данных varchar;
* Birthday – тип данных date.

Столбец IdData содержит идентификатор пользователя, которому соответствует запись в таблице, FullName – ФИО пользователя, Birthday – дату Дня рождения пользователя.

Таблица UsersParams состоит из столбцов:

* Id – тип данных int;
* IdParams – тип данных int;
* ParamsDate – тип данных date;
* UserWeight – тип данных decimal;
* UserHeight – тип данных decimal.

В столбце IdParams хранится идентификатор пользователя, которому соответствует запись в таблице, ParamsDate – дата занесения отчёта физических характеристиках пользователя, UserWeight – масса тела пользователя, UserHeight – рост пользователя.

Таблица Products включает столбцы:

* Id – тип данных int;
* IdAdded – тип данных int;
* ProductName – тип данных varchar;
* CaloriesGram – тип данных decimal;
* ProteinsGram – тип данных decimal;
* FatsGram – тип данных decimal;
* CarbohydratesGram – тип данных decimal;
* FoodCategory – тип данных varchar.

Столбец IdAdded содержит идентификатор пользователя, добавившего продукт в коллекцию, ProductName – название продукта, CaloriesGram ­­– количество калорий на сто грамм продукта, ProteinsGram – количество грамм белков на сто грамм продукта, FatsGram – количество грамм жиров на сто грамм продукта, CarbohydratesGram – количество грамм углеводов на сто грамм продукта, FoodCategory – категория продукта.

Таблица Reports включает столбцы:

* Id – тип данных int;
* IdReport – тип данных int;
* ProductName – тип данных varchar;
* ReportDate – тип данных datetime;
* EatPeriod – тип данных varchar;
* DayGram – тип данных decimal;
* DayCalories – тип данных decimal;
* DayProteins – тип данных decimal;
* DayFats – тип данных decimal;
* DayCarbohydrates – тип данных decimal;

Столбец IdReport содержит идентификатор пользователя, добавившего отчёт о потреблённом в пищу продукте, ProductName – название потреблённого пользователем продукта в пищу, ReportDate – время и дата внесения отчёта пользователем, EatPeriod – название периода приёма пищи, DayGram – количество грамм потреблённого продукта, DayCalories – количество калорий на количество грамм потреблённого продукта, DayProteins – количество грамм белков на количество грамм потреблённого продукта, DayFats – количество грамм жиров на количество грамм потреблённого продукта, DayCarbohydrates – количество грамм углеводов на количество грамм потреблённого продукта.

Скрипты создания базы данных приложения представлены в Приложении A.

## Хранимые процедуры

Хранимая процедура представляет из себя объект базы данных, который содержит в себе набор SQL–инструкций, которые компилируются единожды и хранятся на сервере.

В результате разработки курсового проекта было создано 79 процедур для следующих целей:

1. Выборка всех данных для таблиц;
2. Выборка данных по идентификатору;
3. Выборка данных по значению одного из атрибутов для полей поиска в приложении;
4. Заполнение определённых таблиц на 100 000 строк;
5. Удаление данных из таблиц;
6. Добавление данных в таблицах;
7. Изменение данных в таблицах;
8. Экспорт данных таблицы Products в формате JSON;
9. Импорт данных в формате JSON таблицы Products с предшествующим заполнением данными таблицы FoodCategories.

## Выборка данных из таблиц

Для получения данных из таблиц были написаны следующие процедуры:

DFoodCategoriesSelectId, DFoodCategoriesSelect, DFoodCategoriesSelectAll, DFoodCategoriesSelectCount, DProductsSelect, DProductsSelectAll, DProductsSelectCount, DProductsSelectId, DProductsSelectName, DProductsSelectUser, DProductsSelectUserCount, DReportsSelect, DReportsSelectAll, DReportsSelectCount, DReportsSelectDatePeriod, DReportsSelectDatePeriodCount, DReportsSelectDateUser,

DReportsSelectId, DUsersDataSelect, DUsersDataSelectAll, DUsersDataSelectCount, DUsersDataSelectId, DUsersDataSelectUser, DUsersParamsSelect, DUsersParamsSelectAll, DUsersParamsSelectCount, UsersParamsSelectCountIdParams, DUsersParamsSelectId, DUsersParamsSelectIdParams, DUsersParamsSelectUser, DUsersSelect, DUsersSelectAll, DUsersSelectCount, DUsersSelectId, DUsersSelectLogin.

Процедуры с постфиксом «Count» в названии используются для получения количество строк в соответствующей таблице. Пример подобной процедуры представлен на рисунке 2.2.

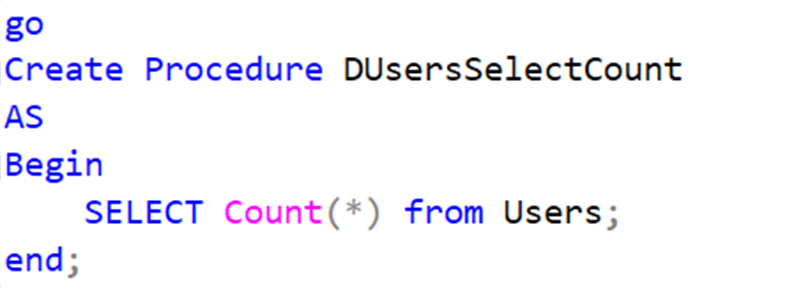


Рисунок 2.2 – Процедура для получения количества строк

Процедуры с постфиксом «All» в названии используются для выборки всех данных из определённой таблицы. Пример подобной процедуры представлен на рисунке 2.3.

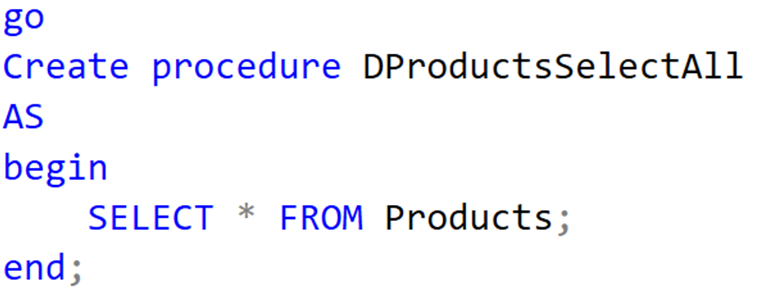


Рисунок 2.3 – Процедура для получения количества строк

Также были созданы процедуры, в постфиксе названия которых используются названия определённых атрибутов таблиц, соответствующих тем полям, по которым производится выборка в процедуре. Пример подобной процедуры представлен на рисунке 2.4.

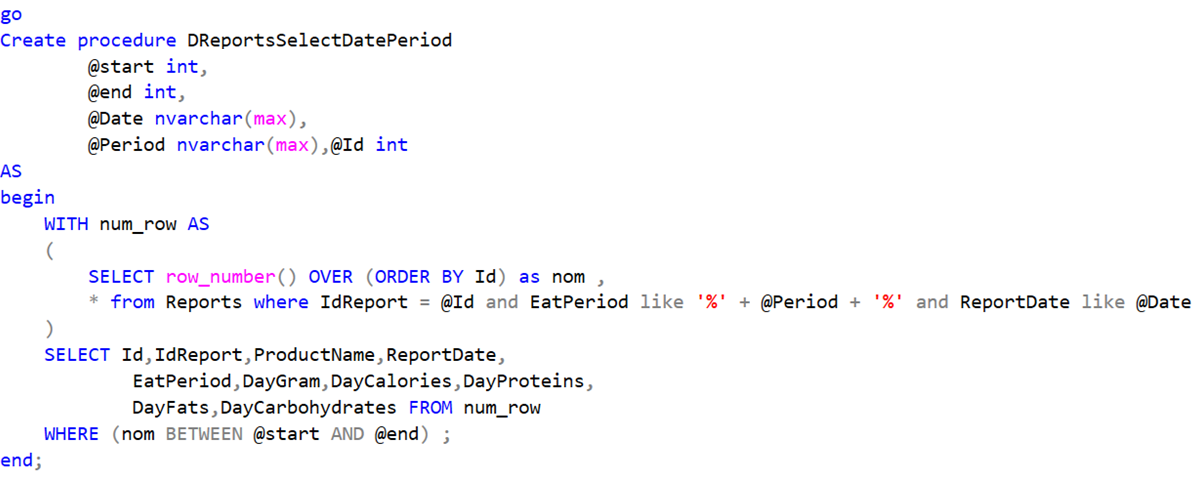


Рисунок 2.4 – Процедура для получения строк по значениям атрибутов

Процедуры с постфиксом «Select» используются для выборки данных из таблиц с использованием встроенной функции row\_number(), позволяющей пронумеровать строки результирующего набора запроса и получить только те строки, значения номеров которых входят в диапазон значений, передаваемых в процедуру в качестве двух числовых параметров. Пример подобной процедуры представлен на рисунке 2.5.

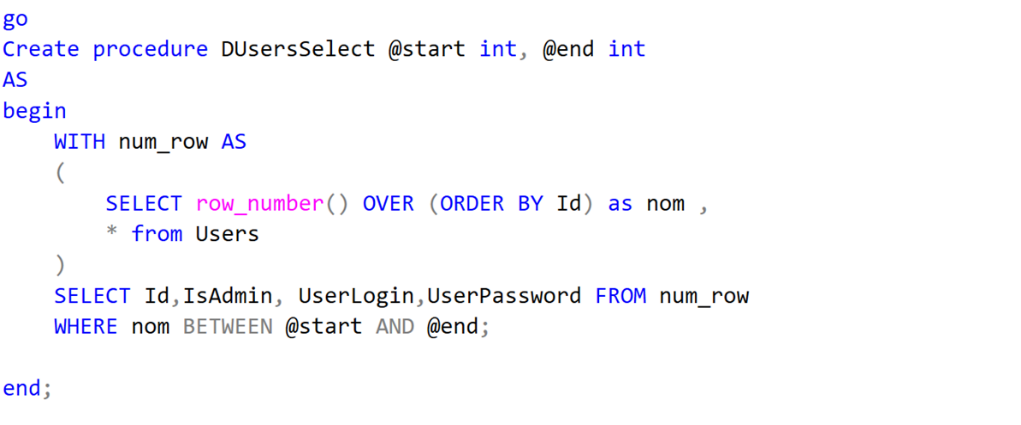


Рисунок 2.5 – Процедура для получения строк в определённом диапазоне

Процедуры для поиска пронумерованных строк со значениями индекса в определённом диапазоне были использованы для реализации встроенной пагинации в разработанном приложении: отображение таблиц, содержащих количество строк, равное 100 000 и более весьма затруднительно и замедляет работу приложения. Поэтому было принято решение о выводе значений результирующего набора по 10 строк на страницу, а также о предоставлении возможности поиска данных по всем полям таблиц для ускорения пользовательской навигации по результирующему набору.

## 2.2.2 Выборка данных по поисковому запросу

Для выборки данных из таблиц по поисковому запросу были разработаны следующие процедуры: DFoodCategoriesSelectCountS, DFoodCategoriesSelectS, DProductsSelectCategoryCountS,DProductsSelectCategoryS, DProductsSelectUserS, DProductsSelectCategoryUserCountS, DReportsSelectS, DProductsSelectCountS, DProductsSelectProductNameCountS, DProductsSelectProductNameS, DProductsSelectS, DProductsSelectCategoryUserS, DProductsSelectUserCountS, DReportsSelectCountS, DUsersDataSelectCountS, DUsersDataSelectS, DUsersParamsSelectCountS, DUsersParamsSelectS, DUsersSelectCountS, DUsersSelectS.

Процедуры с постфиксом «CountS» в названии используются для получения количества строк в таблице, которые соответствуют значению поискового запроса, передаваемого в виде строки в процедуру. Пример подобной процедуры представлен на рисунке 2.6.

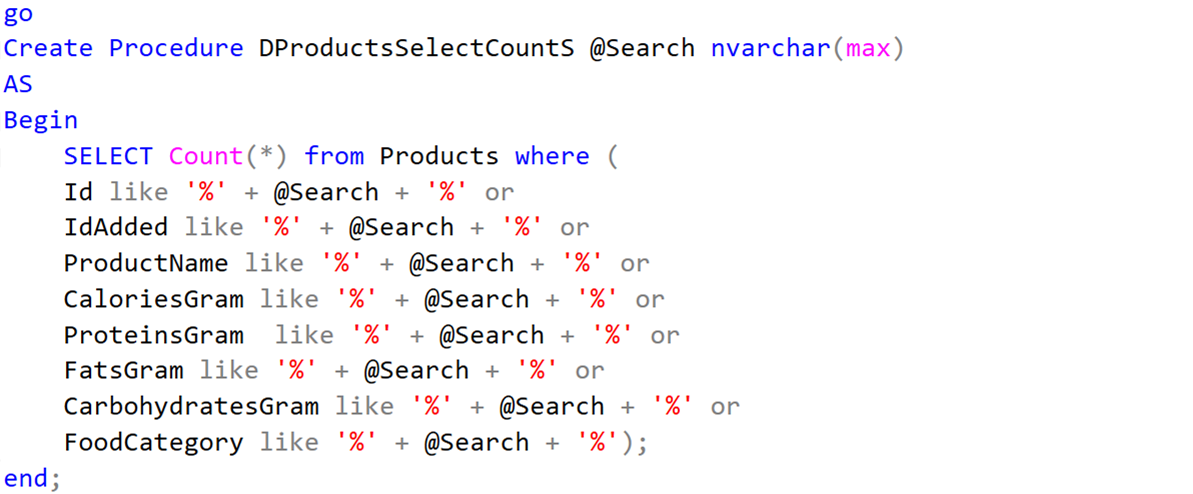


Рисунок 2.6 – Процедура получения количества строк по поисковому запросу

Процедуры с постфиксом «SelectS» в названии используются для получения результирующего набора из таблиц по передаваемому в виде строки в процедуру поисковому запросу. Пример подобной процедуры представлен на рисунке 2.7.

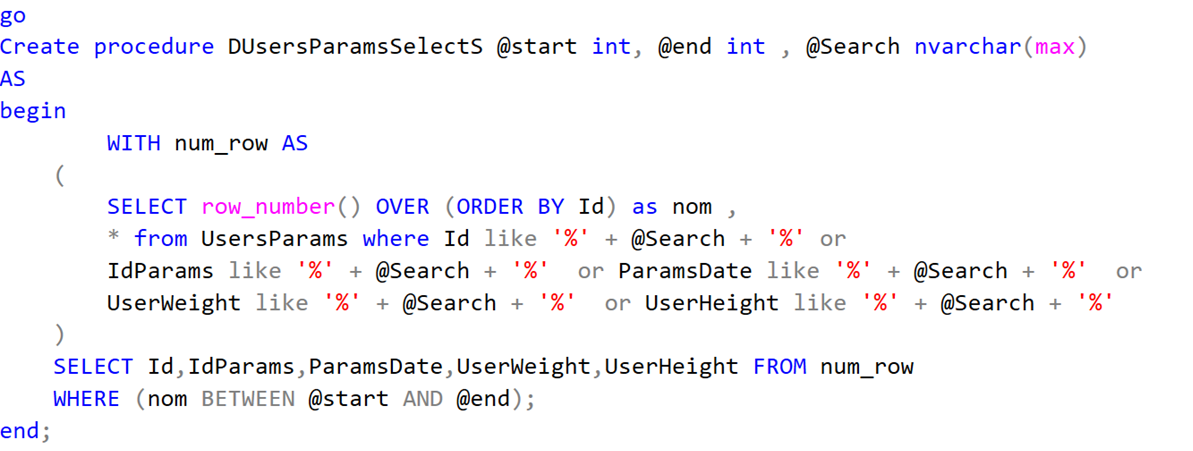


Рисунок 2.7 – Процедура получения строк по поисковому запросу

## 2.2.3 Добавление данных в таблицы

Для добавления данных в таблицы были созданы следующие процедуры: DFoodCategoriesAdd, DProductsAdd, DReportsAdd, DUserAdd, DUsersParamsAdd, DUsersDataAdd. Они были разработаны соответственно для заполнения таблиц: FoodCategories, Products, Reports, Users, UsersParams и UsersData. Пример подобной процедуры представлен на рисунке 2.8.

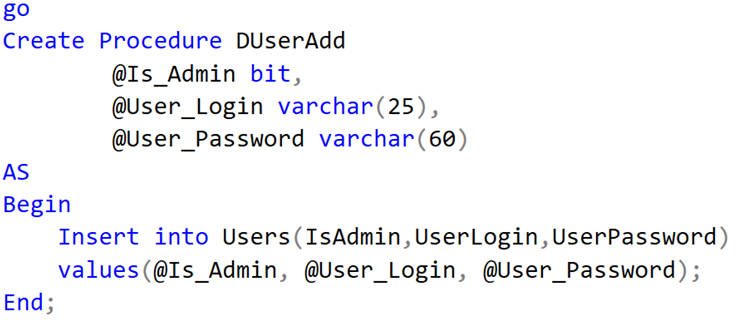


Рисунок 2.8 – Процедура добавления записи в таблицу

## 2.2.4 Удаление данных из таблиц

Для удаления данных из таблиц были созданы следующие процедуры: DFoodCategoriesDelete, DProductsDelete, DReportsDelete, DUserDelete, DUsersParamsDelete, DUsersDataDelete. Они были разработаны соответственно для заполнения таблиц: FoodCategories, Products, Reports, Users, UsersParams и UsersData. Пример подобной процедуры представлен на рисунке 2.9.

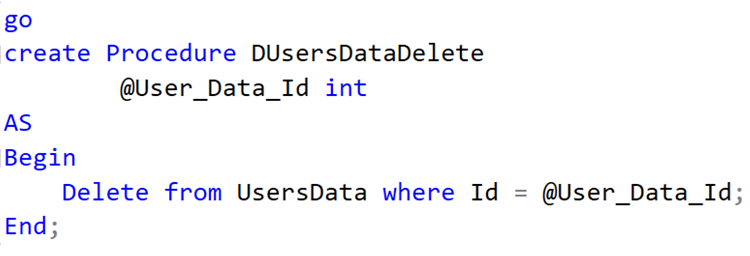


Рисунок 2.9 – Процедура удаления записи из таблицы

## 2.2.4 Изменение данных в таблицах

Для изменения данных в таблицах были созданы следующие процедуры: DFoodCategoriesUpdate, DProductsUpdate, DReportsUpdate ,DUserUpdate, DUsersParamsUpdate, DUsersDataUpdate. Они были разработаны соответственно для заполнения таблиц: FoodCategories, Products, Reports, Users, UsersParams и UsersData. Пример подобной процедуры представлен на рисунке 2.10.

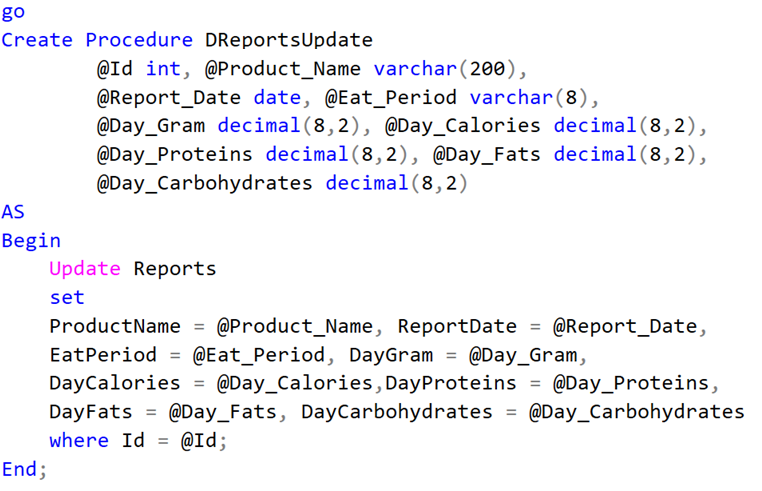


Рисунок 2.10 – Процедура изменения записи в таблице

## 2.2.5 Экспорт и импорт данных в формате JSON

Для экспорта данных таблицы Products в формате JSON была разработана процедура ExportProductsJSON. Код данной процедуры представлен на рисунке 2.11.

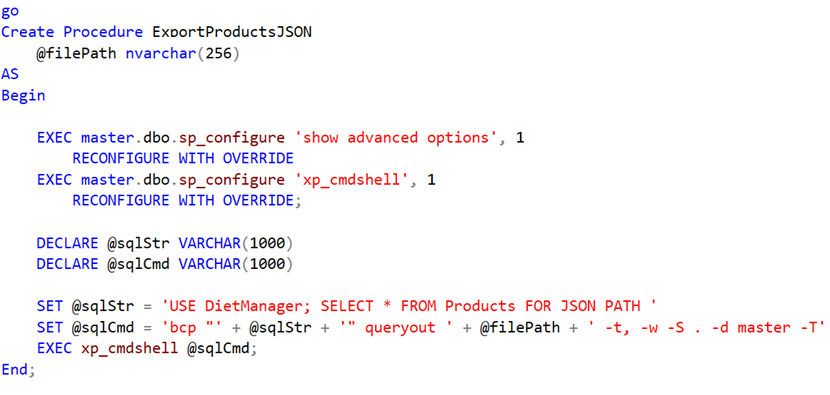


Рисунок 2.11 – Процедура экспорта данных в формате JSON

Для формирования JSON в select запросе используется конструкция FOR JSON PATH. Для вывода в файл была использована расширенная хранимая процедура xp\_cmdshell. Фрагмент JSON строки представлен в Приложении Б.

Для импорта данных в таблицу Products из файла формата JSON, была разработана процедура ImportProductsFromJSON. Для получения JSON строки и последующего разбора со вставкой используется конструкция FROM OPENROWSET совместно с параметром BULK. Процедура ImportProductsFromJSON представлена ниже на рисунке 2.12.

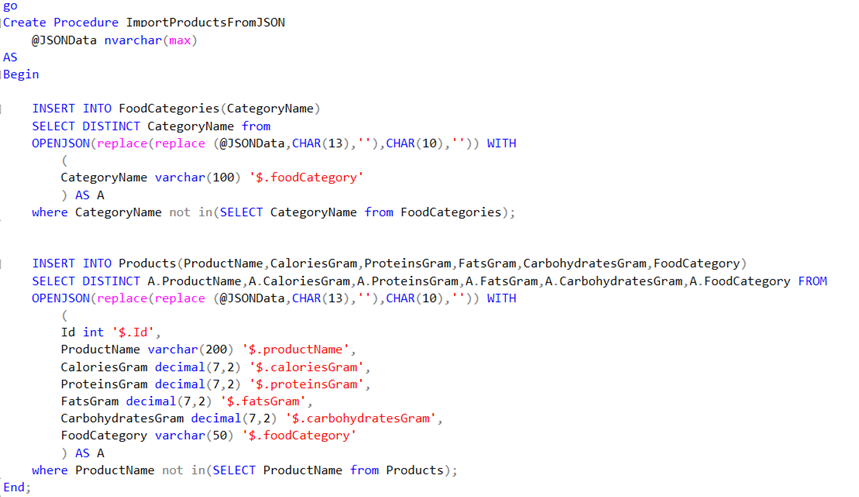


Рисунок 2.12 – Процедура импорта данных в формате JSON

Особенностью данной функции является то, что, если категория продуктов, указанная в одной из записей таблицы Products в JSON строке будет отсутствовать в таблице FoodCategories, отсутствующее значение категории будет предварительно добавлено в таблицу категорий продуктов.

## 2.3 Индексы

В общее число разработанных объектов базы данных также входят индексы, используемые для ускорения процесса выборки данных из таблиц. Таблицы в базе данных могут иметь большое количество строк, которые хранятся в произвольном порядке, и их поиск по заданному критерию путём последовательного просмотра таблицы строка за строкой может занимать много времени. В связи с необходимостью ускорения формирования результирующего набора из таблиц, было принято решение разработать 48 индекса для таблиц: FoodCategories, Users, UsersData, UsersParams, Reports и Products.

## 2.4 Многомерные структуры данных

Одной из целей написания курсового проекта было изучение технологии создания многомерных структур данных OLAP.

OLAP (Online Analytical Processing)- это система аналитической обработки данных. Она предназначена для подготовки отчетов, построения прогностических сценариев и выполнения статистических расчетов на базе больших информационных массивов, имеющих сложную структуру.

Одним из способов аналитической обработки сложных данных является построение многомерных OLAP-кубов. OLAP-куб – многомерный массив данных, как правило, разреженный и долговременно хранимый, используемый в OLAP. Может быть реализован на основе универсальных реляционных СУБД или специализированным программным обеспечением.

В данном случае OLAP-кубы были построены на основе информации, хранящейся в реляционной базе данных, используемой для разрабатываемого приложения.

## 2.4.1 Измерения OLAP-куба

Было разработаны три измерения, факты из которых использовались для анализа нескольких отобранных мер.

Измерения имеют названия Products, Reports и UsersParams, созвучные с названиями таблиц, на основе которых строятся, и содержащие факты таблиц, используемые в последующем построении OLAP-куба.

На рисунке 2.13 изображена структура измерения Products.

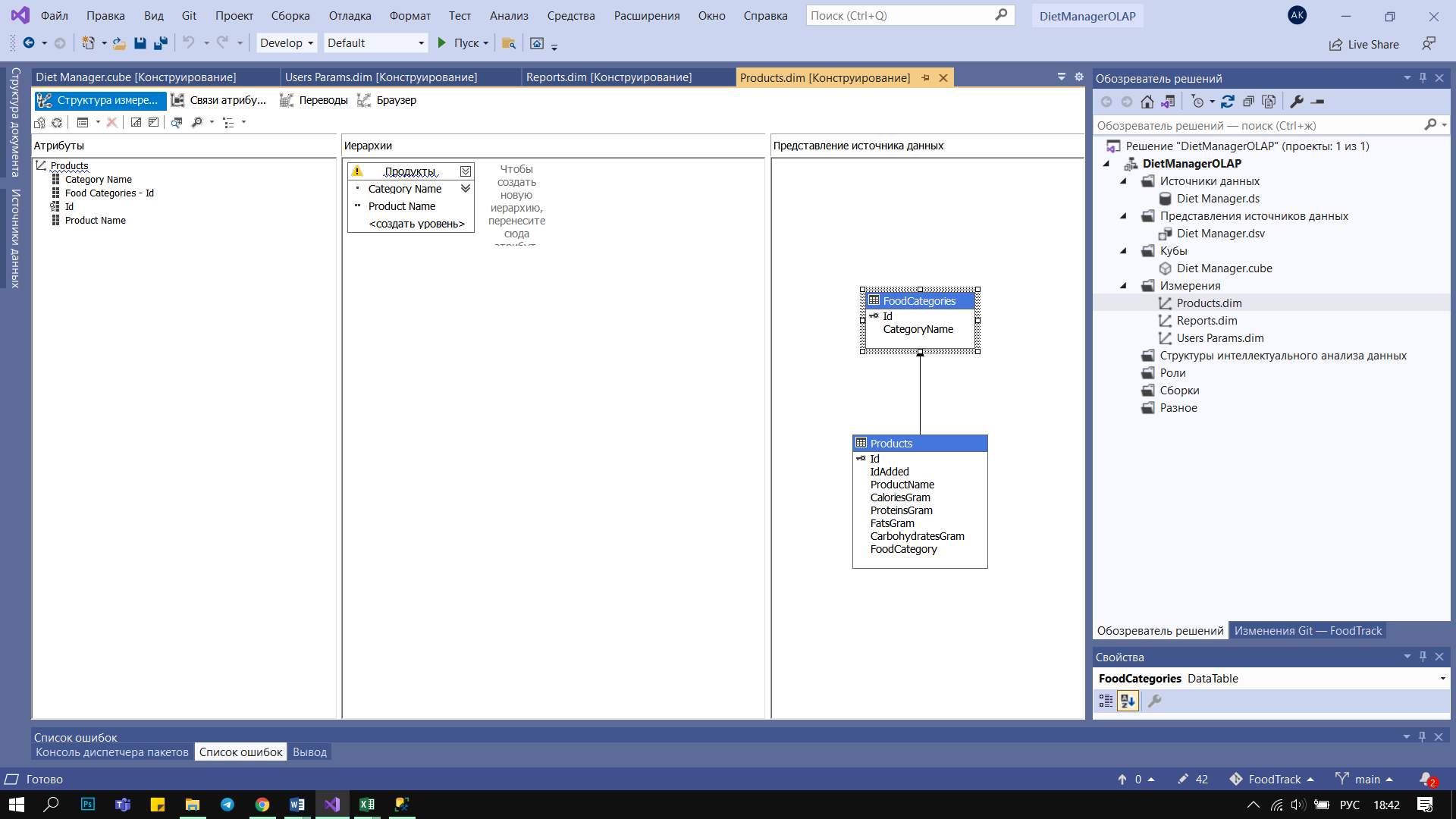


Рисунок 2.13 – Измерение Products

На основе данного измерения была разработана иерархия продуктов по категориям, фрагмент которой представлен на рисунке 2.14.

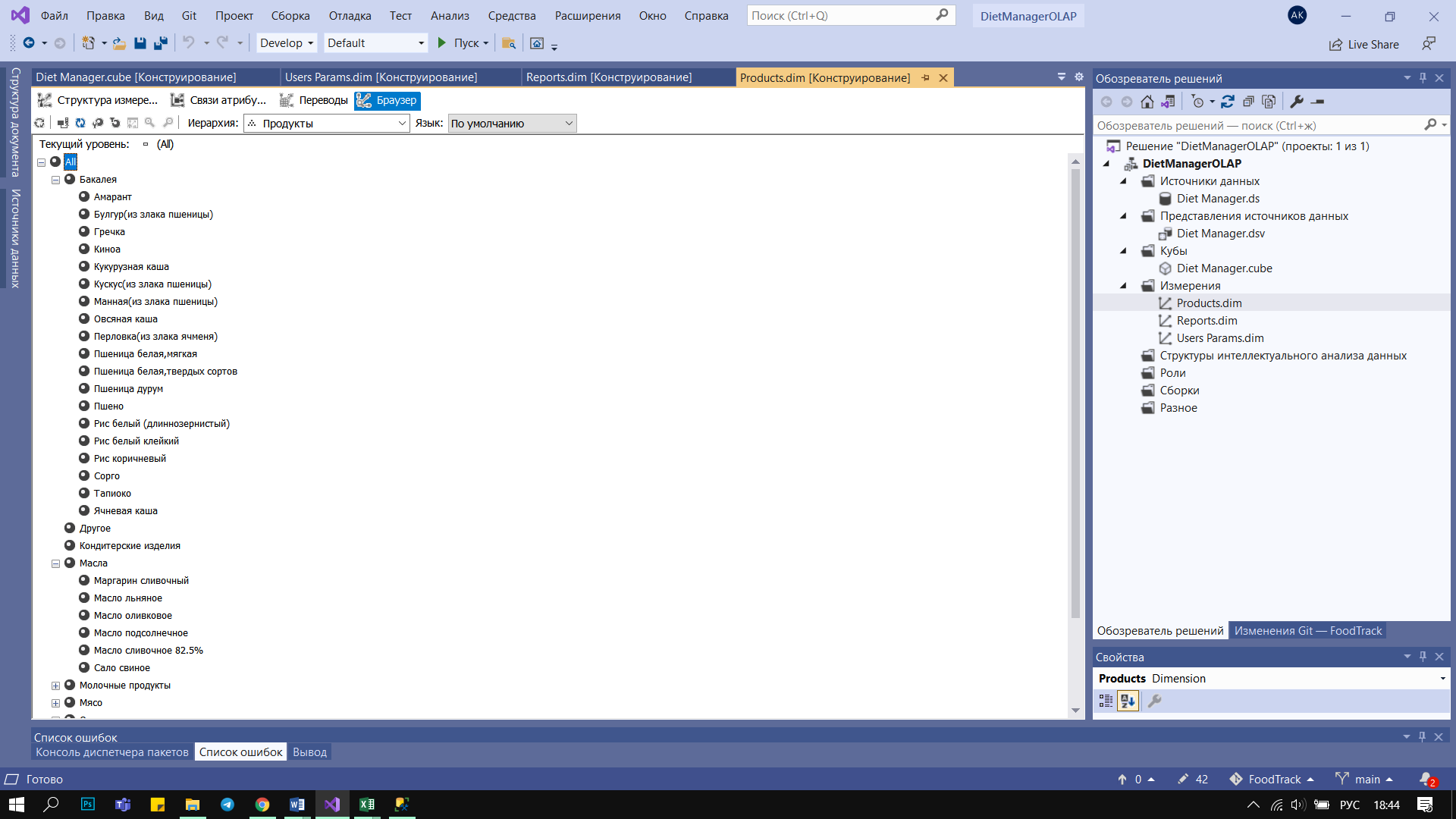


Рисунок 2.14 – Иерархия продуктов по категориям

Измерение Reports использовалось для формирования иерархии продуктов по приёмам пищи. На рисунке 2.15 изображена структура измерения Reports.

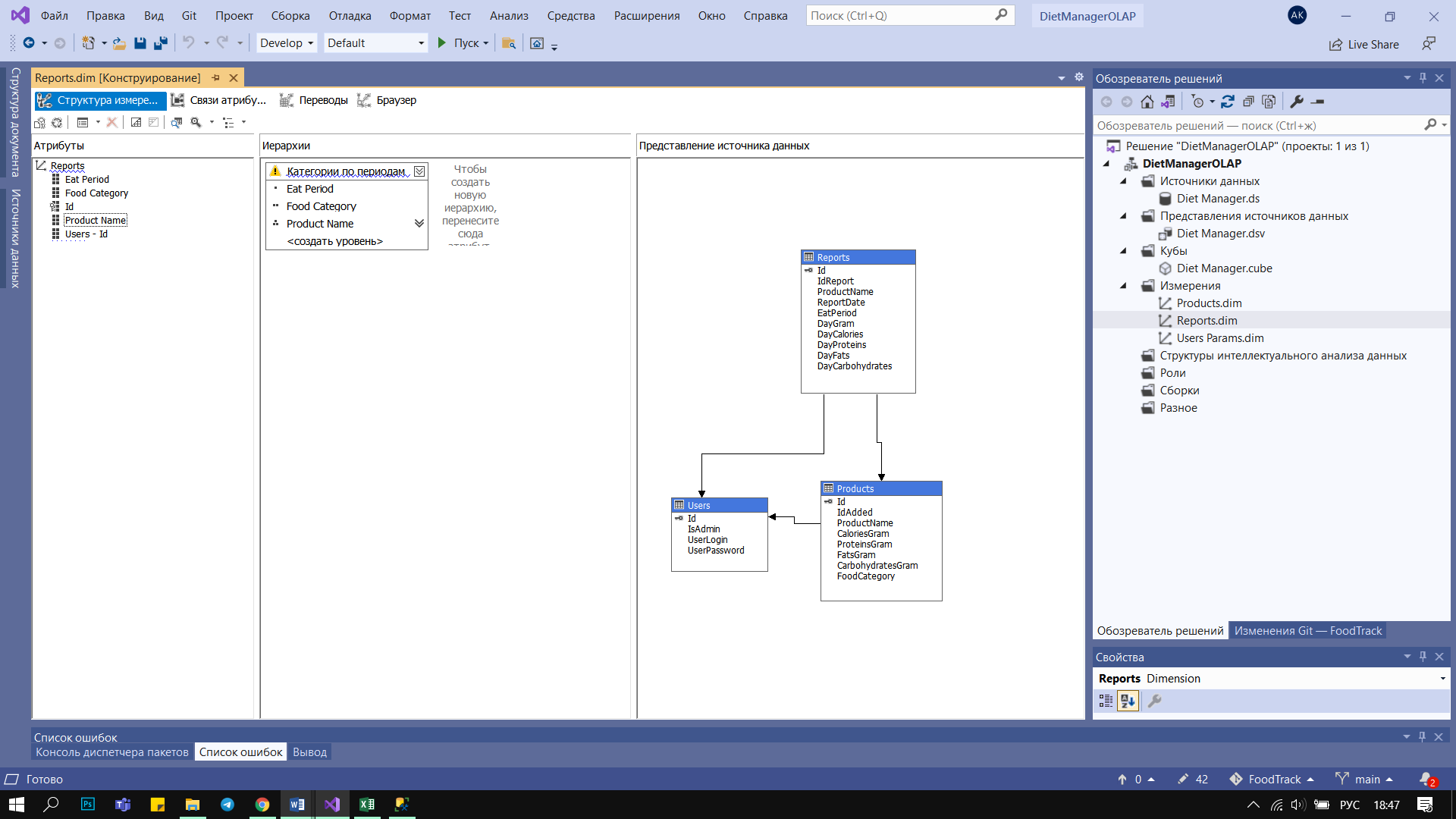


Рисунок 2.15 – Измерение Reports

На основе данного измерения была разработана иерархия категорий продуктов и названий продуктов по приёмам пищи, фрагмент которой представлен на рисунке 2.16.

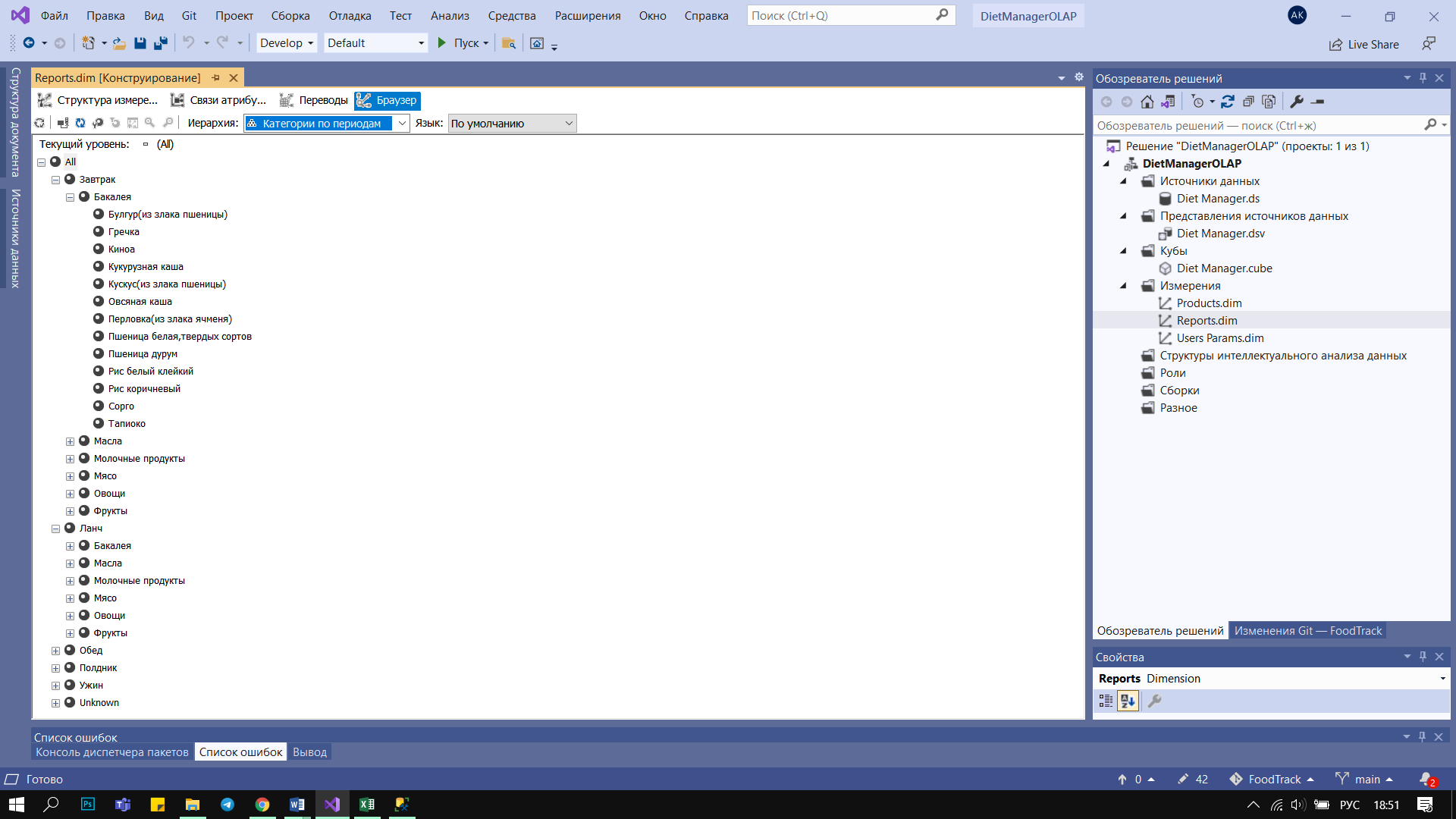


Рисунок 2.16 – Иерархия категорий продуктов и названий продуктов по периодам приём пищи

## 2.4.2 OLAP-куб

На основе разработанных измерений был создан OLAP-куб под названием ReportsAnalysis, позволяющий анализировать данные об отчётах приёмов пищи пользователями по категориям потребляемых продуктов и самих продуктов соответственно. Структура OLAP–куба представлена на рисунке 2.17.

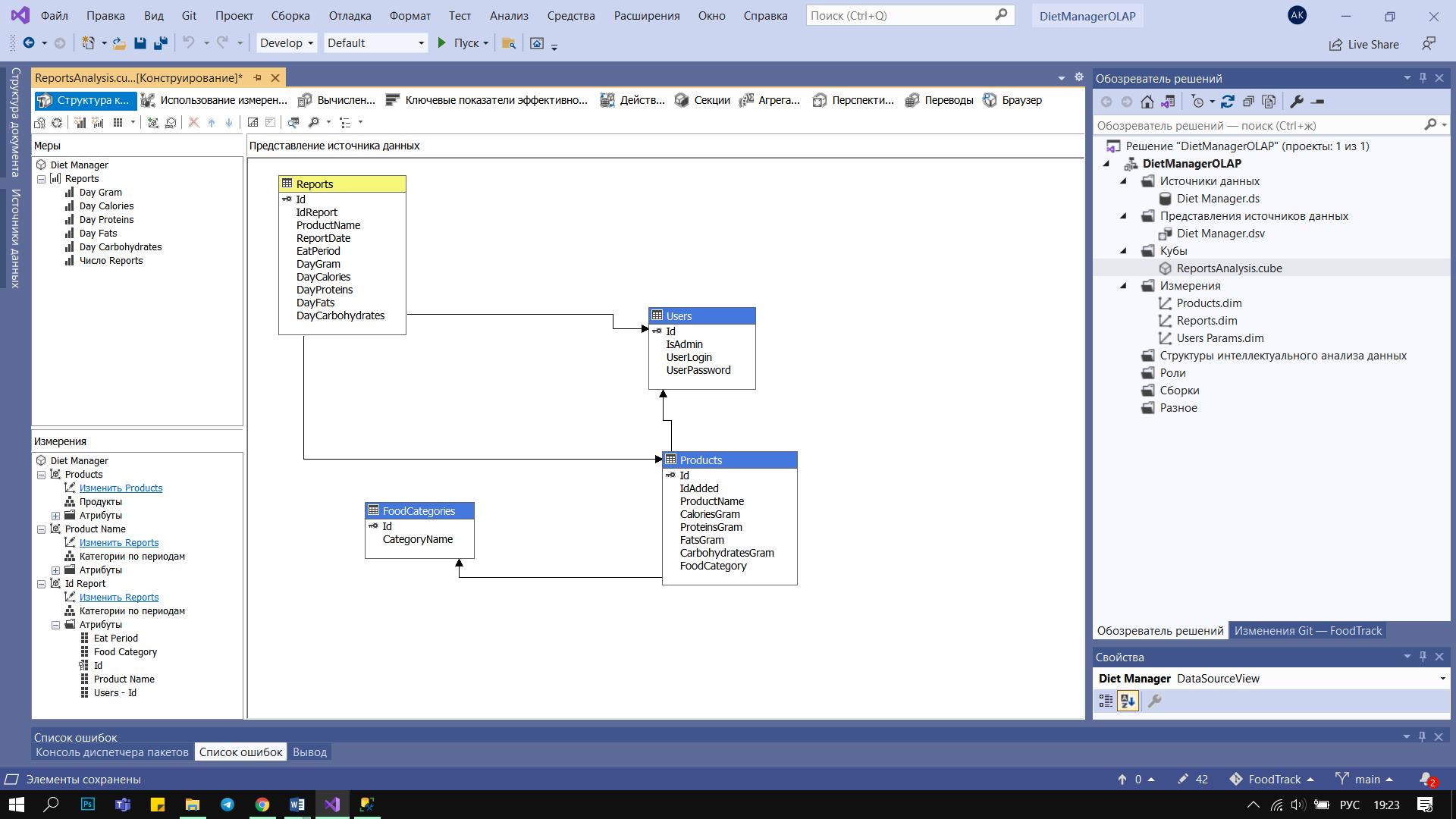


Рисунок 2.17 – Структура OLAP-куба ReportsAnalysis

После построение куба был сформирован запрос для анализа запросов по категориям и потребляемым продуктам. Результат запроса отображён на рисунке 2.18.

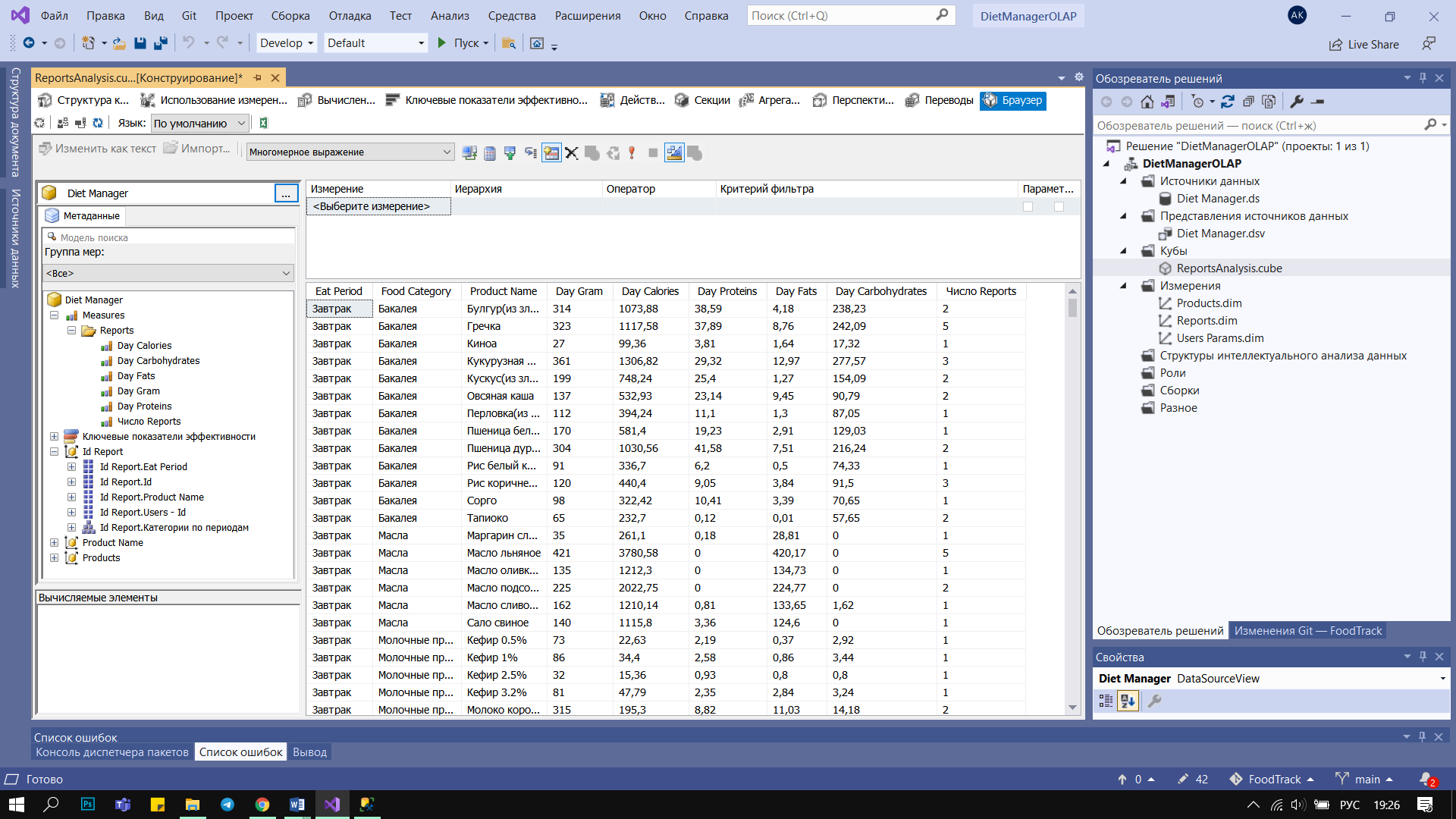


Рисунок 2.18 – Результат запроса к OLAP-кубу

Для удобства отображения результатов запроса к OLAP-кубу можно использовать Microsoft Excel. Результат запроса в Microsoft Excel изображён на рисунке 2.19.

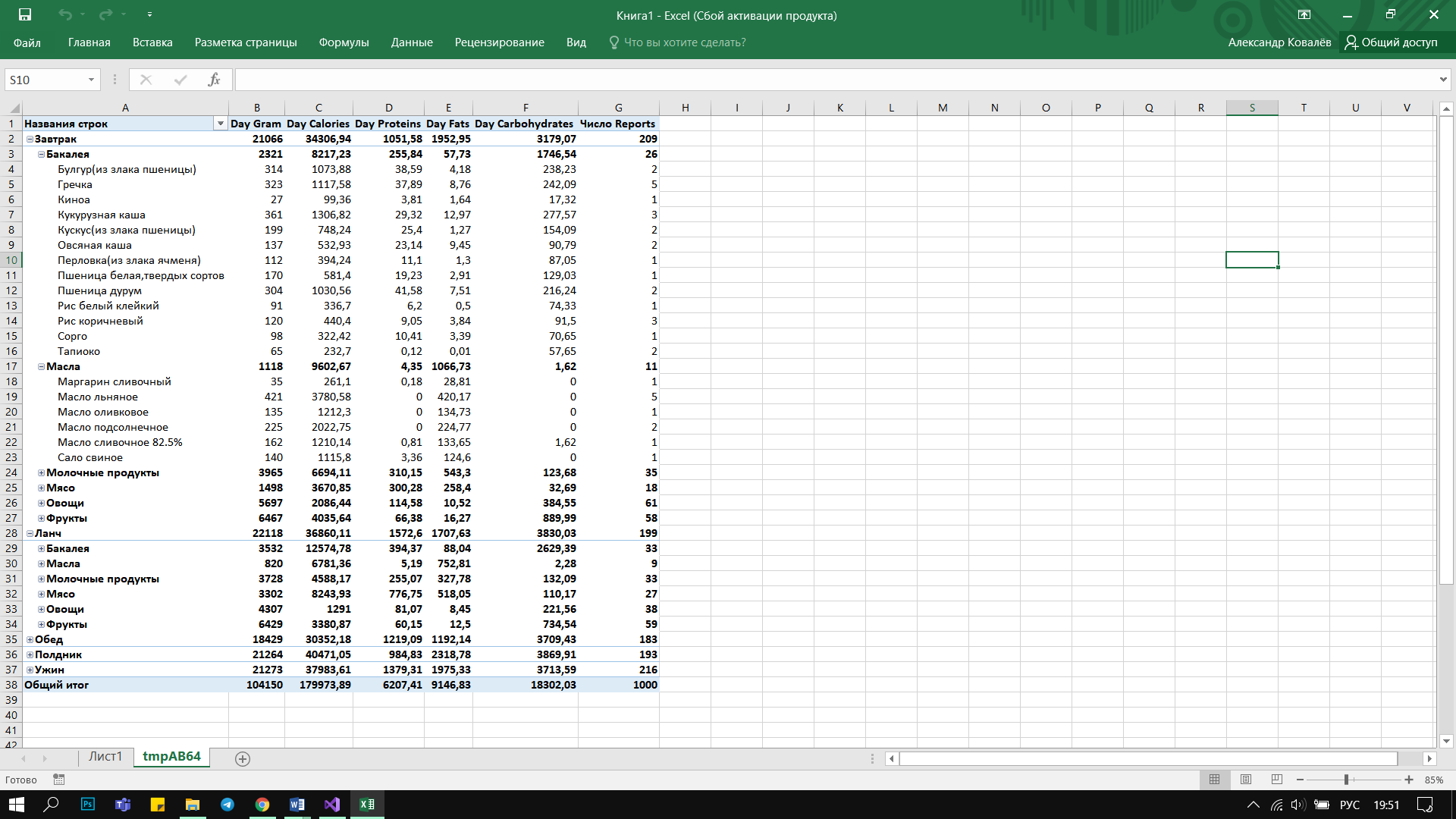


Рисунок 2.19 – Результат запроса к OLAP-кубу

# Обоснование технических приемов программирования

При создании программного средства был выбран язык программирования JAVA с фреймворком Spring. Взаимодействие c БД осуществляется с использованием технологии Java Persistence API.

Java Persistence API — спецификация API Java EE, предоставляет возможность сохранять в удобном виде Java-объекты в базе данных. JPA реализует концепцию ORM.

Схема взаимодействия приложения с базой данных представлена на рисунке 3.1.

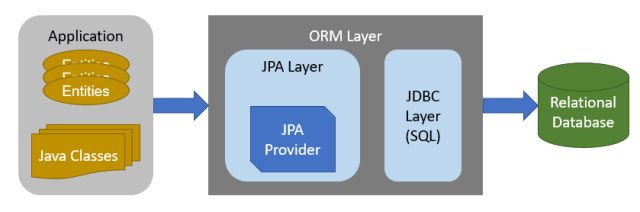


Рисунок 3.1 – Схема взаимодействия приложения с БД

Для подключения к базе данных со стороны приложения на сервере необходимо провести конфигурацию подключения к БД. Фрагмент кода конфигурации подключения предоставлен на рисунке 3.2.

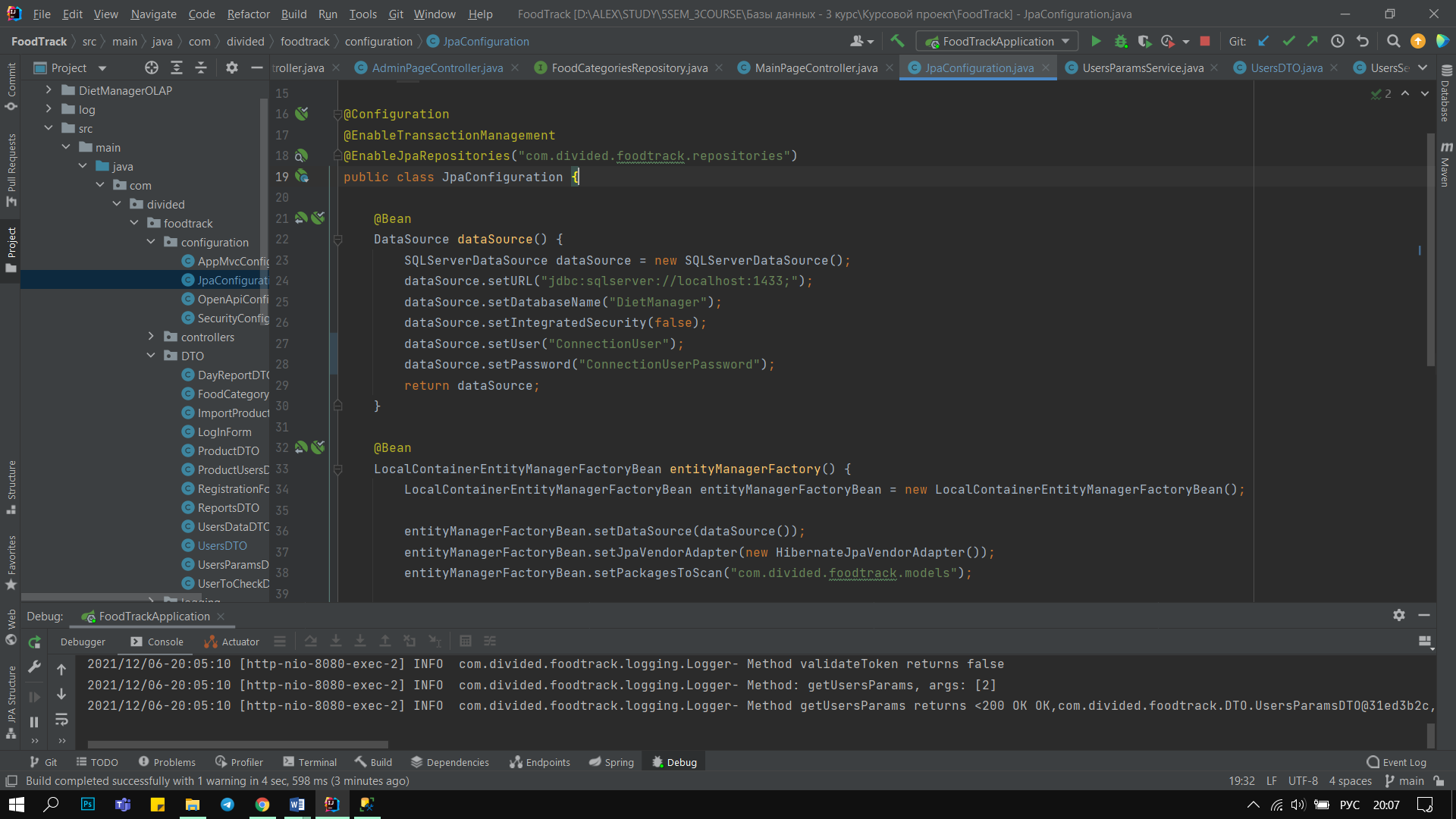


Рисунок 3.2 – Конфигурация подключения к БД

Функционал взаимодействия с БД на стороне сервера реализован при помощи разработанных классов-репозиториев, содержащих методы, осуществляющие вызов хранимых процедур базы данных. Пример реализации одного из репозиториев представлен на рисунке 3.3.

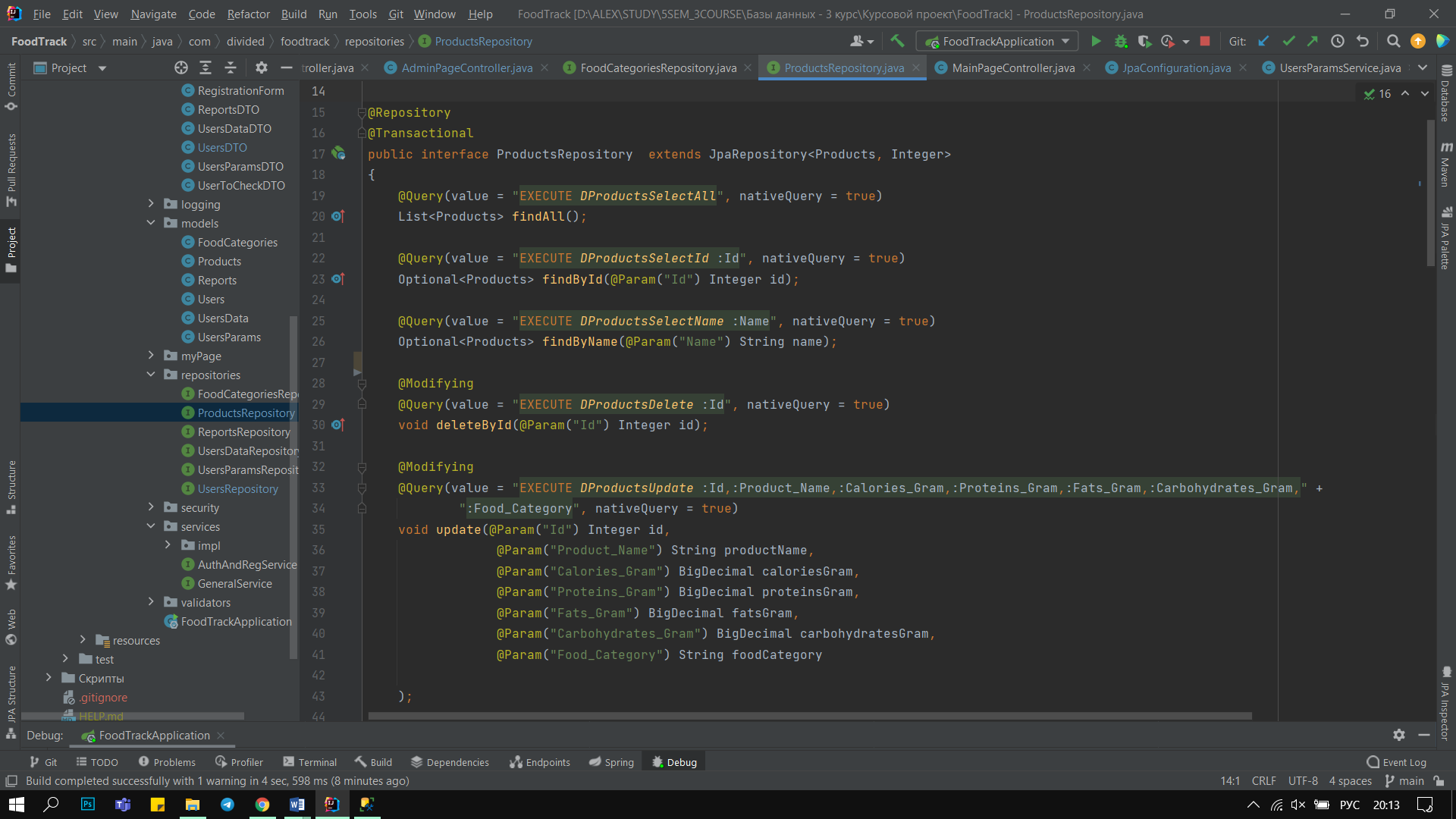


Рисунок 3.3 – Реализация репозитория для взаимодействия с таблицей Products

Использование технологии Java Persistence API приложение должно содержать определение классов-моделей, соответствующих таблицам используемой базы данных. Фрагмент реализации класса-модели для таблицы Reports представлен на рисунке 3.4.



Рисунок 3.4 – Реализация класса-модели для взаимодействия с таблицы Reports

# Тестирование, проверка работоспособности и анализ данных

Первая страница, требующая проверки работоспособности – это страница авторизации. Необходимо ввести логин и пароль пользователя. Страница авторизации изображена на рисунке 4.1.

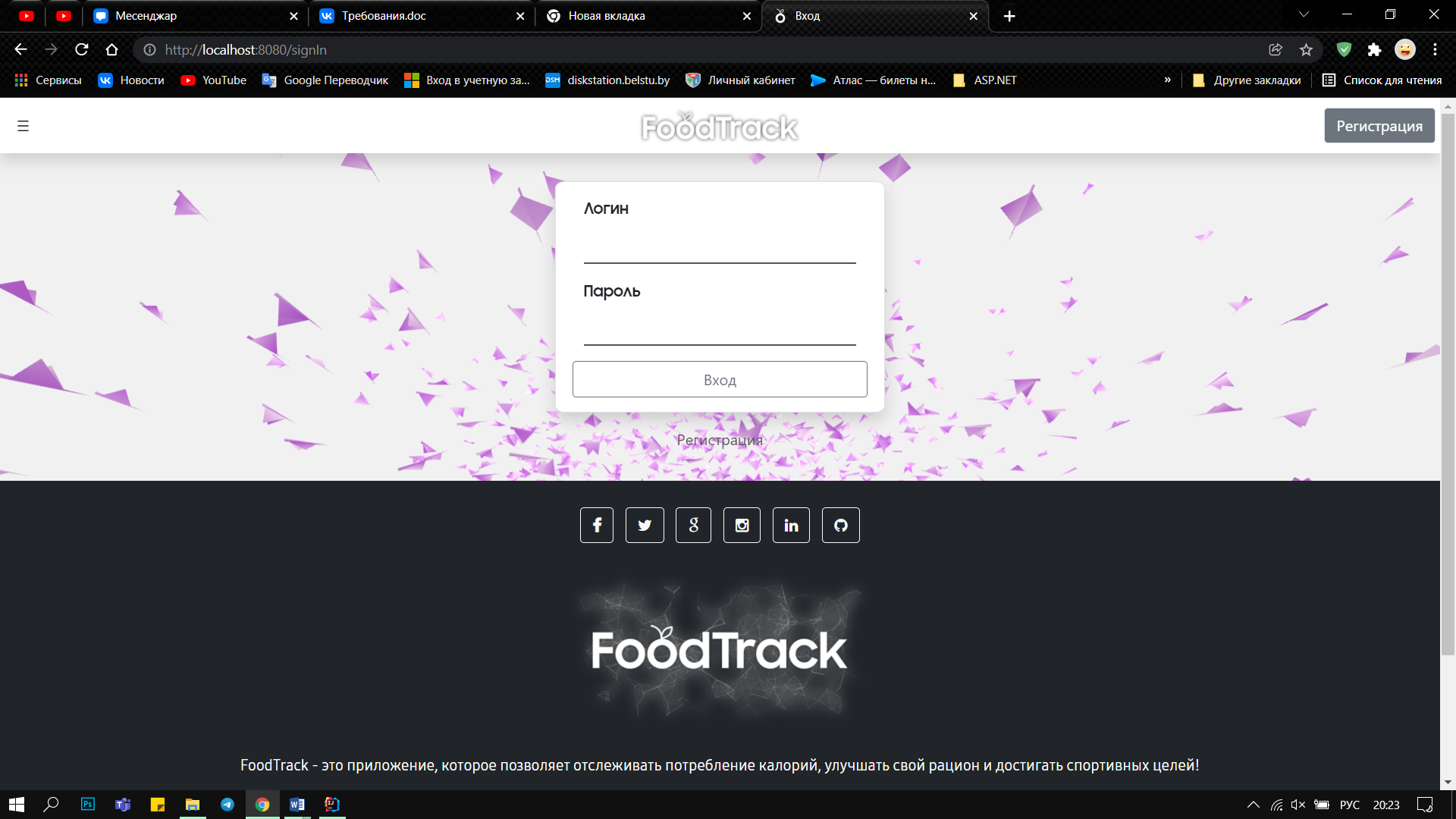


Рисунок 4.1 – Страница авторизации

При вводе правильного логина и пароля происходит переход на главную страницу приложения. При вводе некорректных данных выводится сообщение об ошибке и предложение ввести другие данные в поля логина и пароля.

Страница пользователя разделена на несколько вкладок. Первая вкладка имеет название «Приёмы пищи» и используется для взаимодействия пользователя с записями о приёмах пищи. На данной вкладке предоставлена возможность изменять и удалять записи, а также просматривать информацию о приёмах пищи по датам и периодам приёма пищи. Страница изображена на рисунке 4.2.

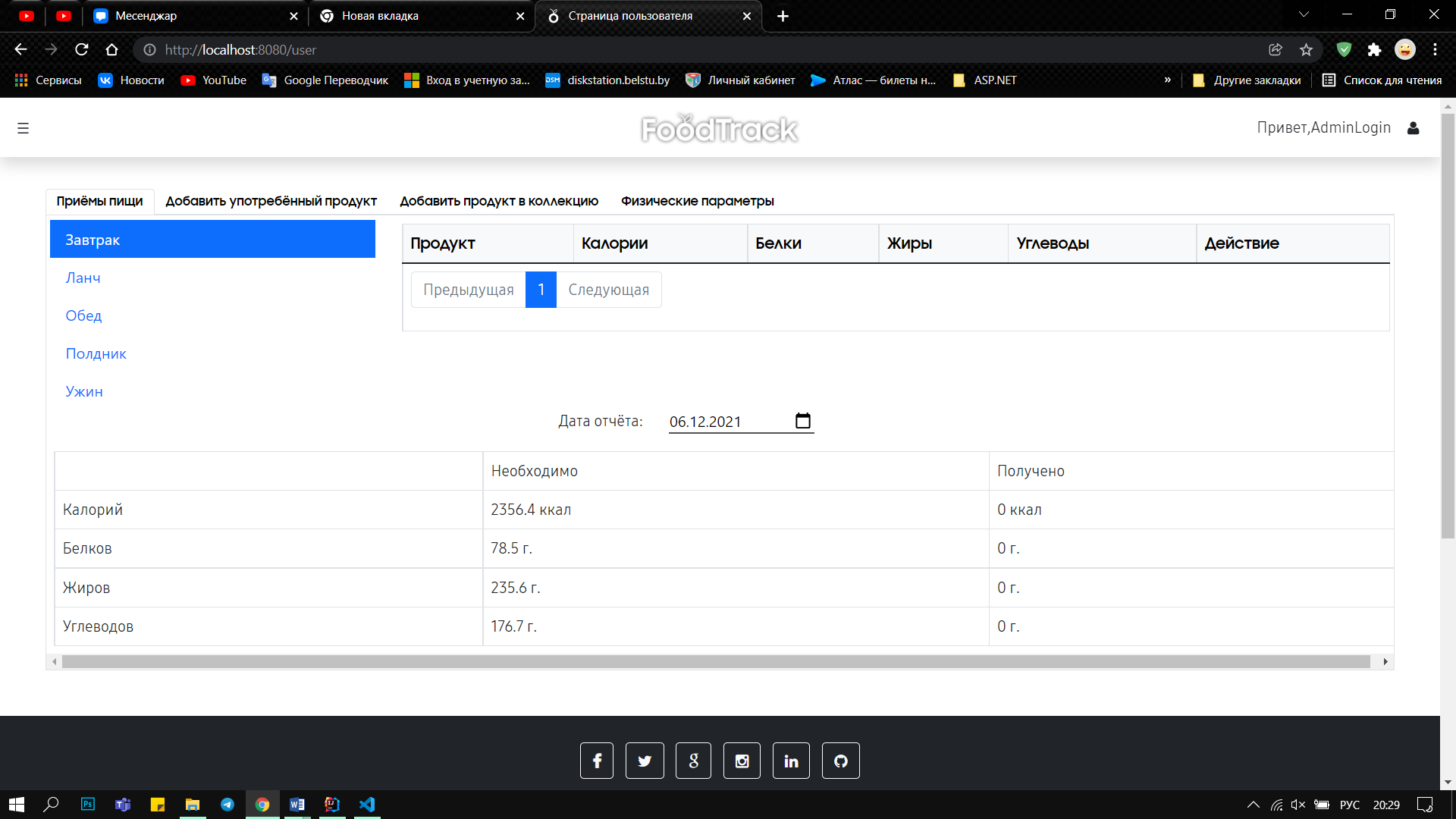


Рисунок 4.2 – Страница приёмов пищи

Для получения данных на момент определённой даты использует поле «Дата отчёта». Для получения данных о приёме пищи по определённому периоду используется навигационное меню в левой части страницы. В нижней части страницы отображено необходимое и потреблённое количество питательных веществ пользователя.

После добавления данных при помощи меню на вкладке «Добавить употреблённый продукт» он отображается в таблице на вкладке «Приёмы пищи». Пользователь может изменить или удалить запись. Пример окна изменения записи представлен на рисунке 4.3.

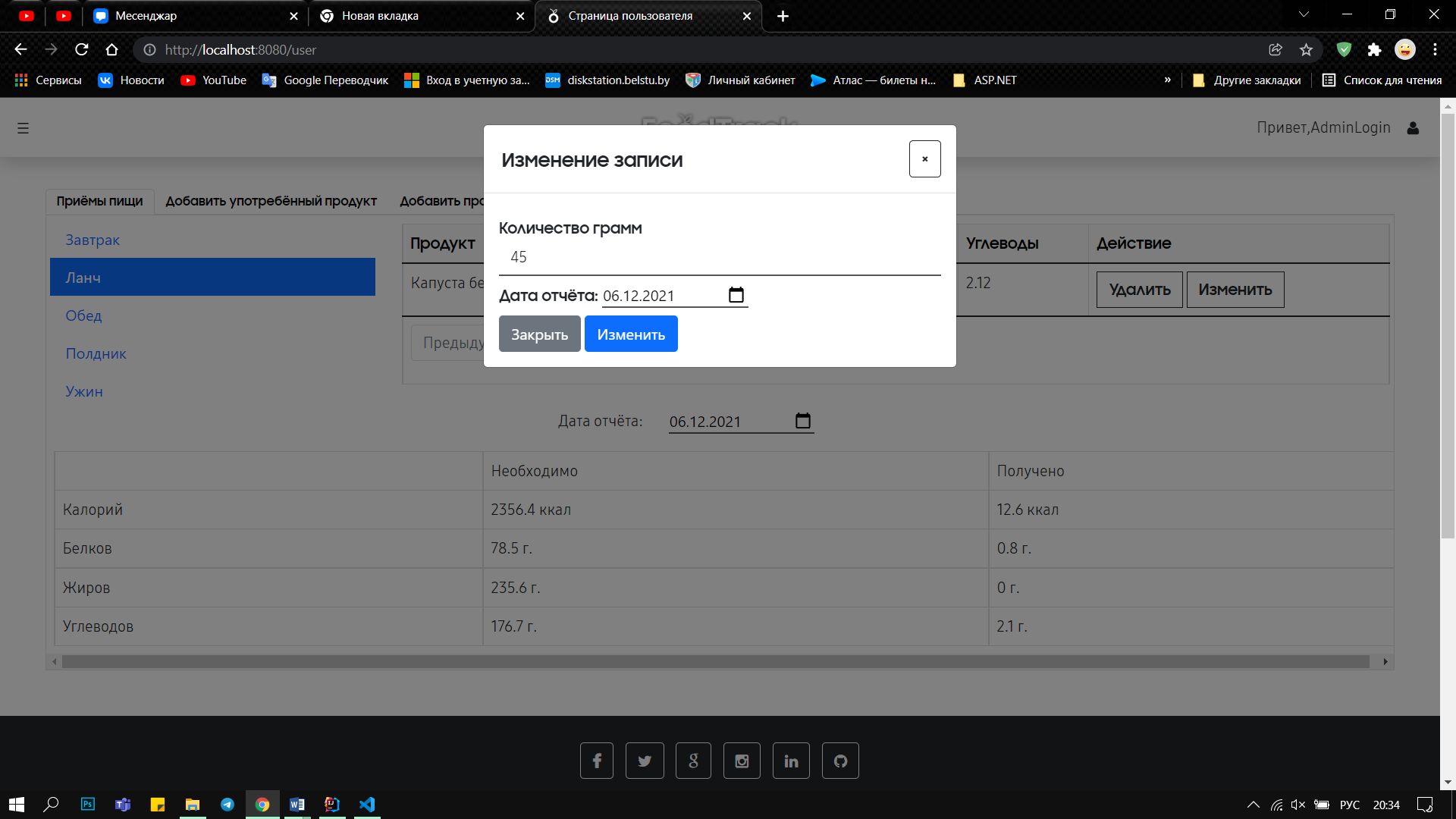


Рисунок 4.3 – Окно изменения записи

Вкладка «Добавить употреблённый продукт» предоставляет функционал добавления отчёта о приёме пищи. Содержимое вкладки изображено на рисунке 4.4.

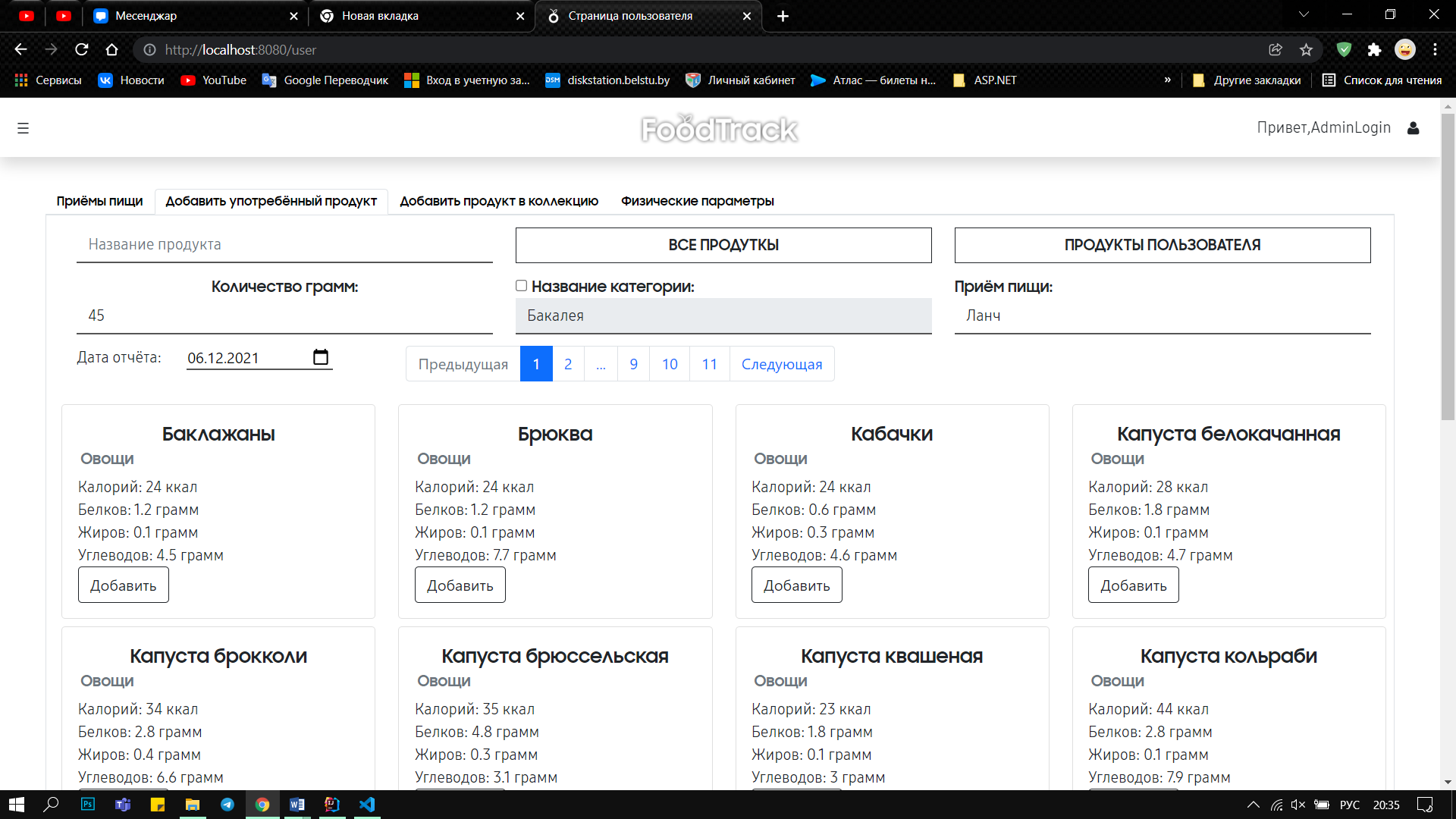


Рисунок 4.4 – Страница добавления

Коллекцию продуктов можно сортировать по категориям продукта. Также можно осуществлять поиск по имени продукта. Присутствует возможность вывести на страницу либо все продукты, либо только продукты пользователя. Для удобства навигации по коллекции продуктов на странице присутствует возможность использования навигации. Пример результата поиска по имени и категории продукта представлен на рисунке 4.5.

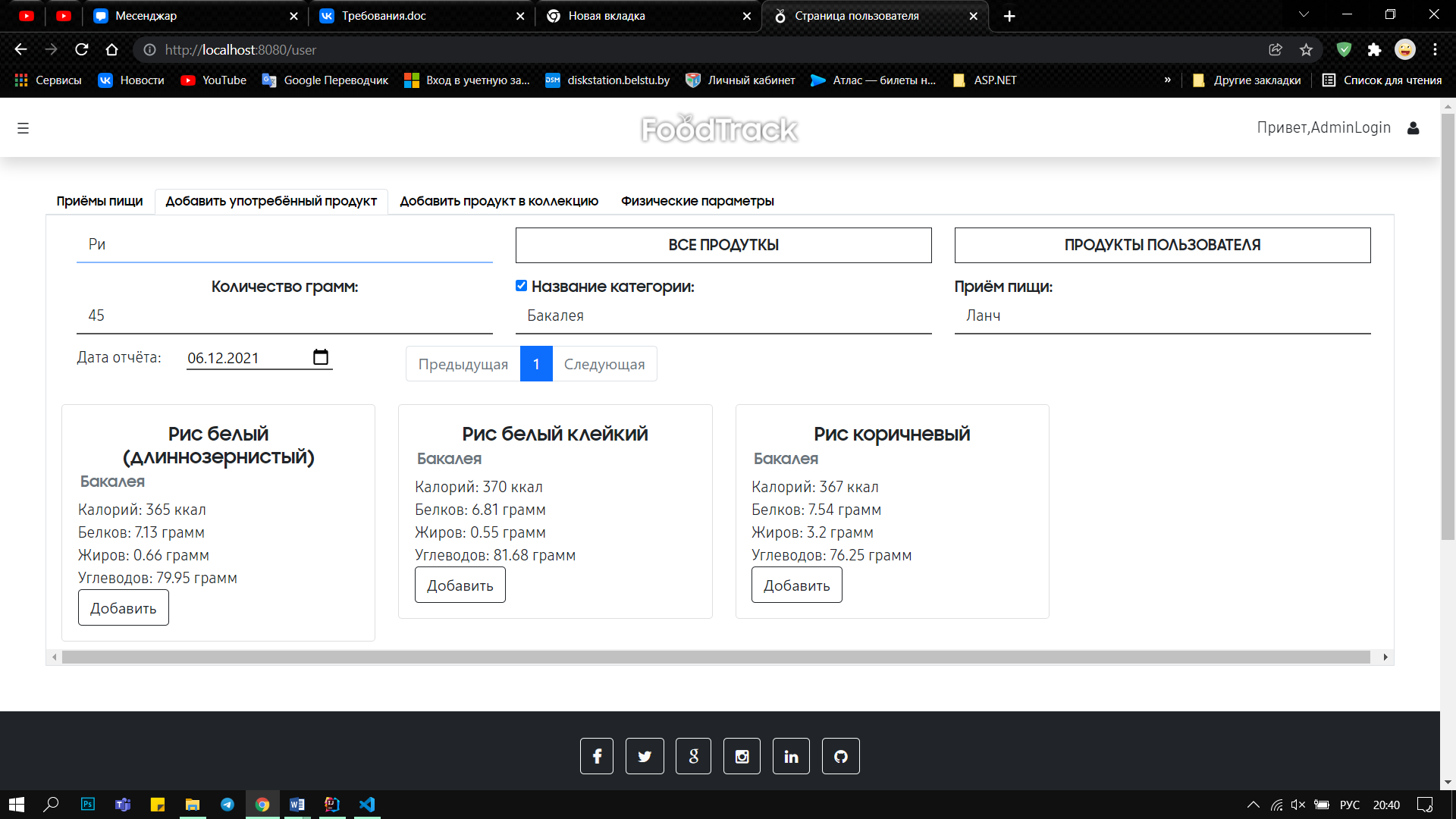


Рисунок 4.5 – Результат поиска

При нажатии на кнопку «Добавить» на одном из элементов коллекции продуктов и после ввода значения количество грамм потребляемого продукта, отчёт будет помещён в таблицу.

На вкладке «Добавить продукт в коллекцию» представлен функционал для добавления пользователем продукта в общую коллекцию продуктов, который в дальнейшем может быть использован всеми пользователями приложения для добавления отчётов. Содержимое вкладки изображено на рисунке 4.6.

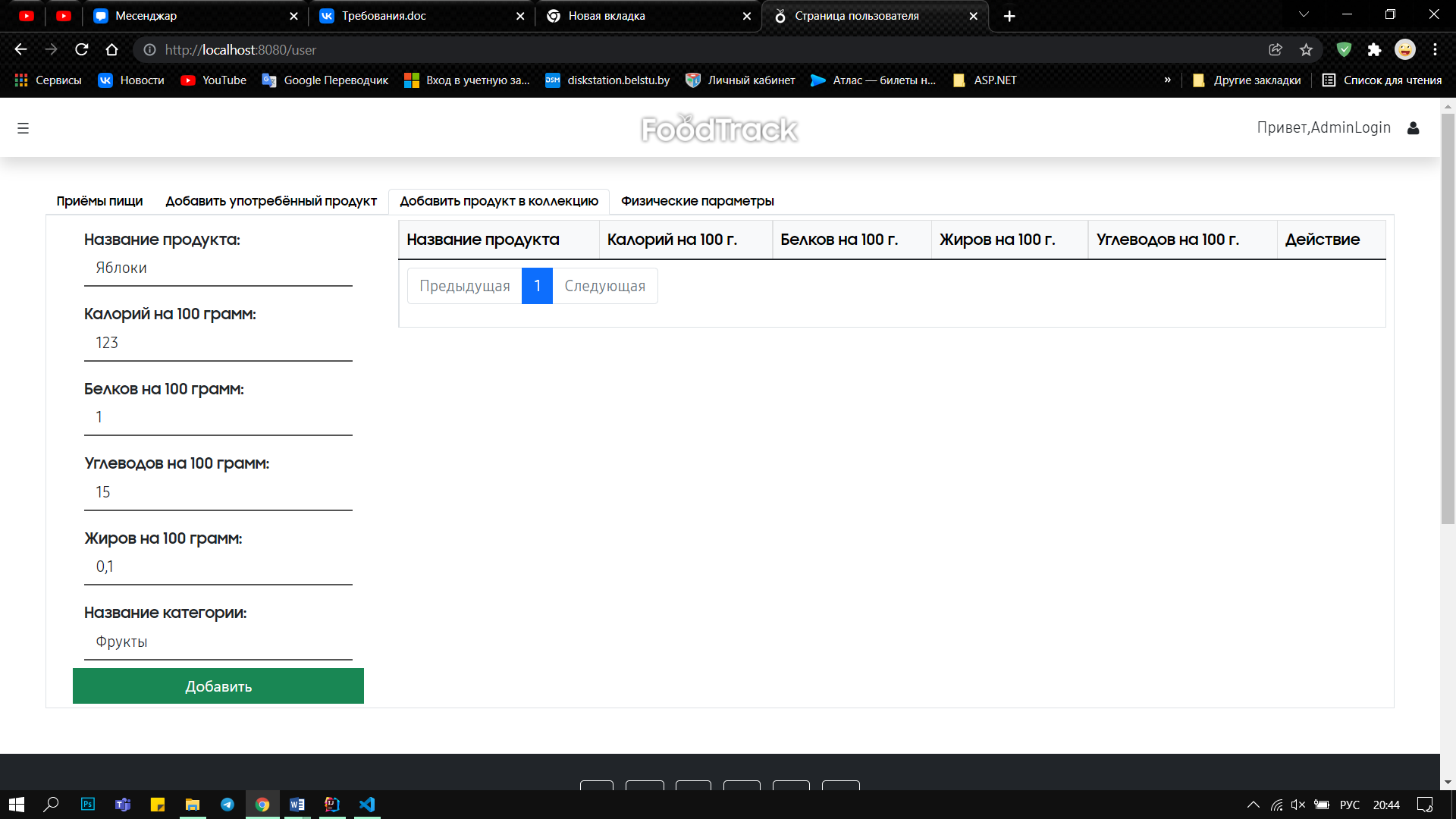


Рисунок 4.6 – Результат поиска

При вводе данных, нарушающих ограничения целостности базы данных, пользователю выводится сообщения о необходимости изменить значения полей. Пример отображаемого уведомления представлено на рисунке 4.7.

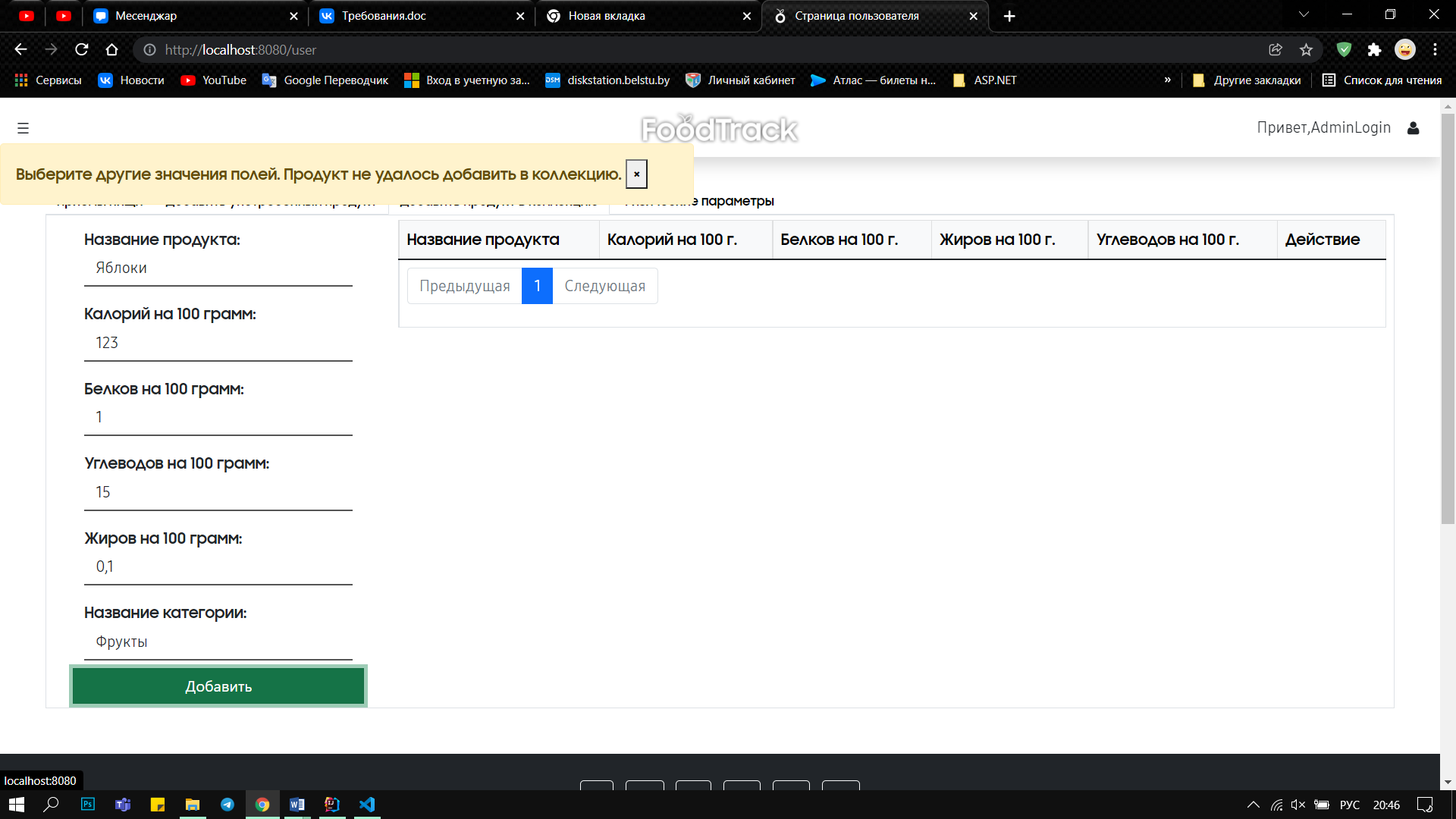


Рисунок 4.7 – Уведомление об ошибке

При успешном добавлении продукта в коллекцию запись отобразится в таблице и пользователю будет выведено соответствующее сообщение. Пользователь может изменить либо удалить запись. Окно изменении записи представлено на рисунке 4.8.

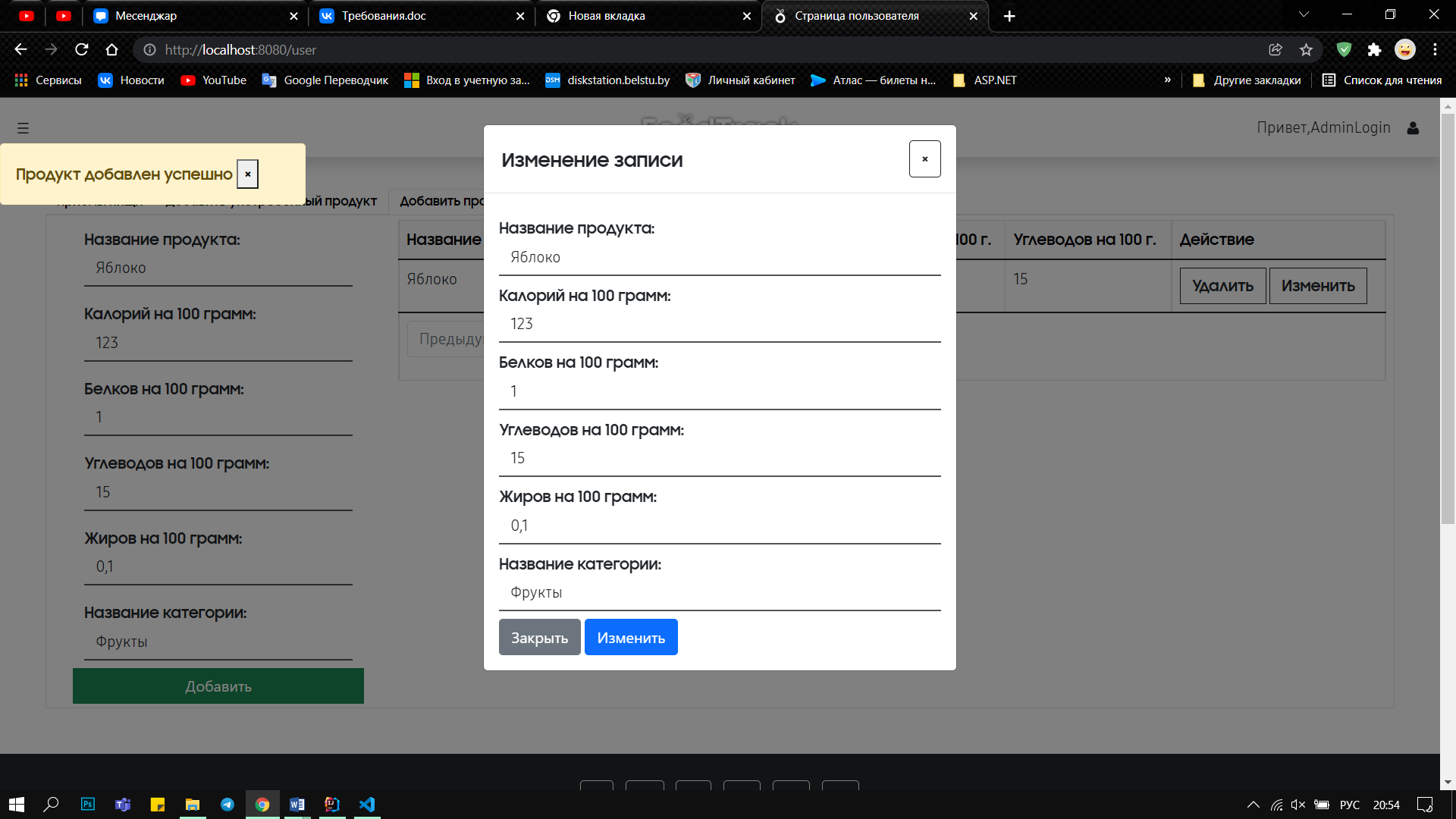


Рисунок 4.8 – Окно изменения продукта

На вкладке «Физические параметры» располагается форма для добавления данных о физических параметрах пользователя и таблица параметров пользователя. При добавлении записи в таблицу пользователю выводится соответствующее уведомление. Содержимое вкладки представлено на рисунке 4.9.

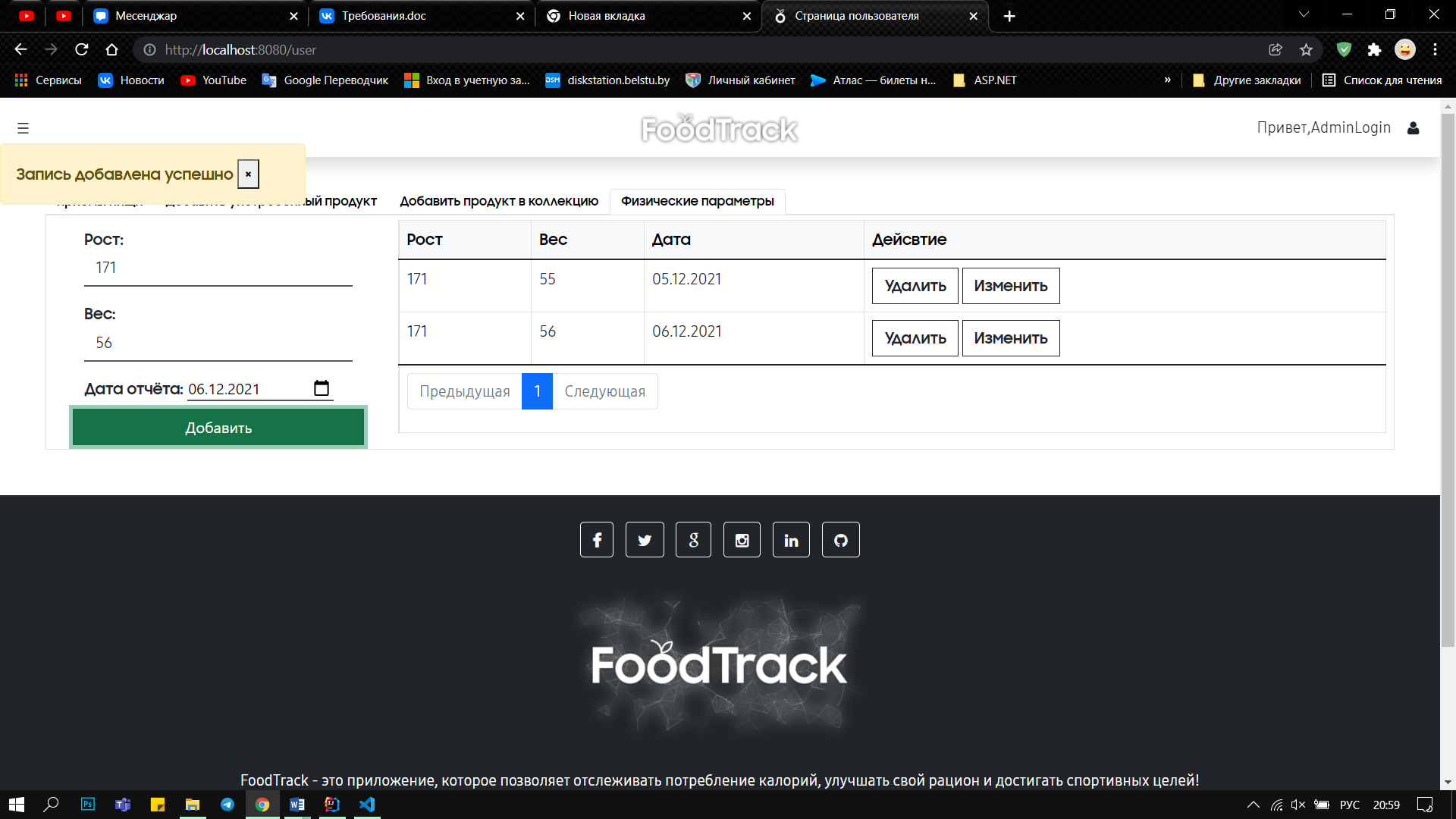


Рисунок 4.9 – Страница физических параметров

Окно изменения записи о физических параметрах представлено на рисунке 4.10.

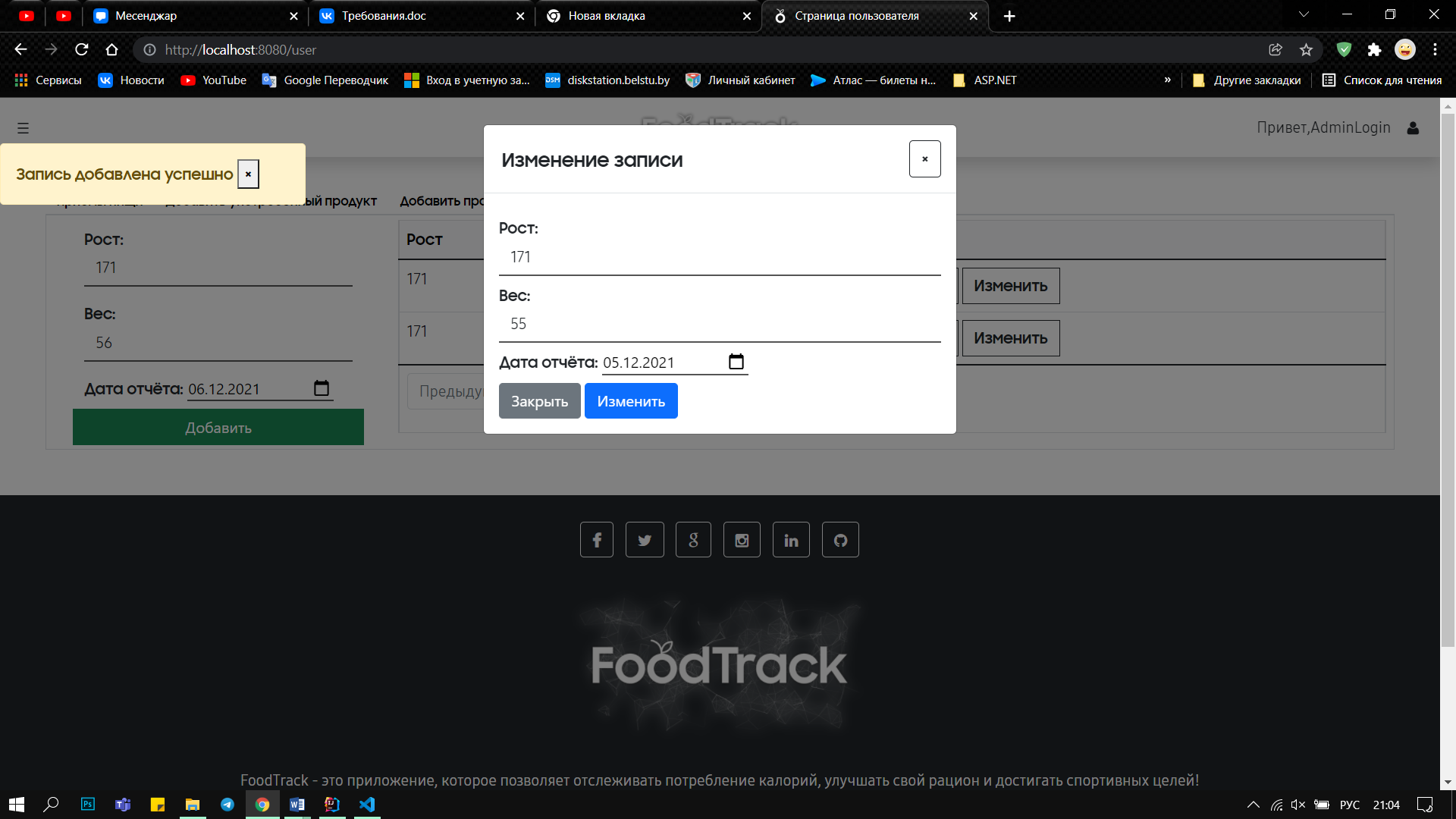


Рисунок 4.10 – Окно изменения записи о физических параметрах

Страница Администратора содержит навигационное меню. На каждой вкладке навигационного меню находится таблица, панель постраничной навигации по записям таблицы, поле поиска и, по необходимости, кнопку добавления записи в таблицу. Таблица FoodCategories представлена на рисунке 4.11.

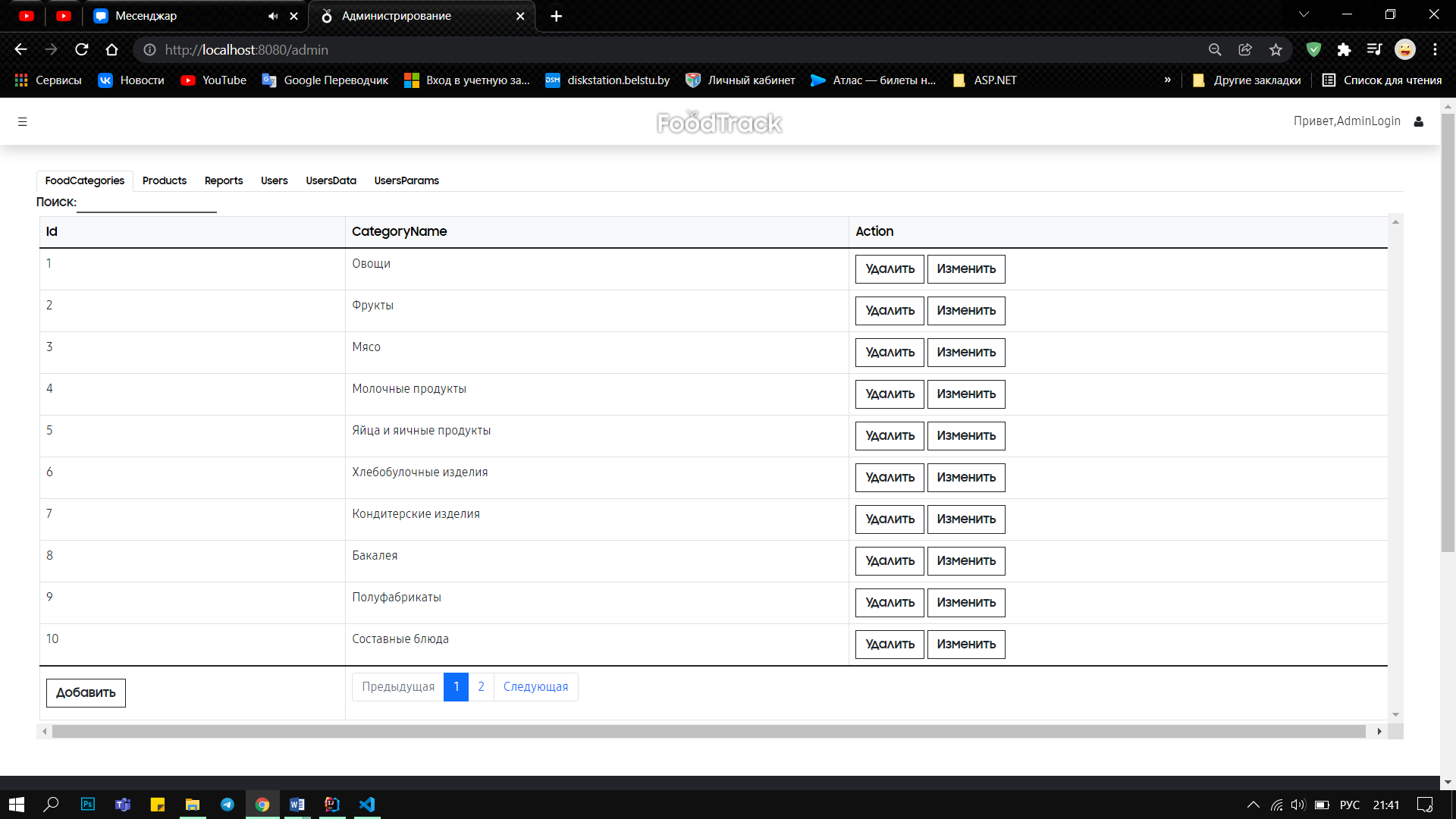


Рисунок 4.10 – Таблица FoodCategories

При вводе неверных данных в одном из полей любой из форм изменения записи таблиц выводится сообщение о необходимости ввода верных данных. Пример уведомления об ошибке представлен на рисунке 4.11.

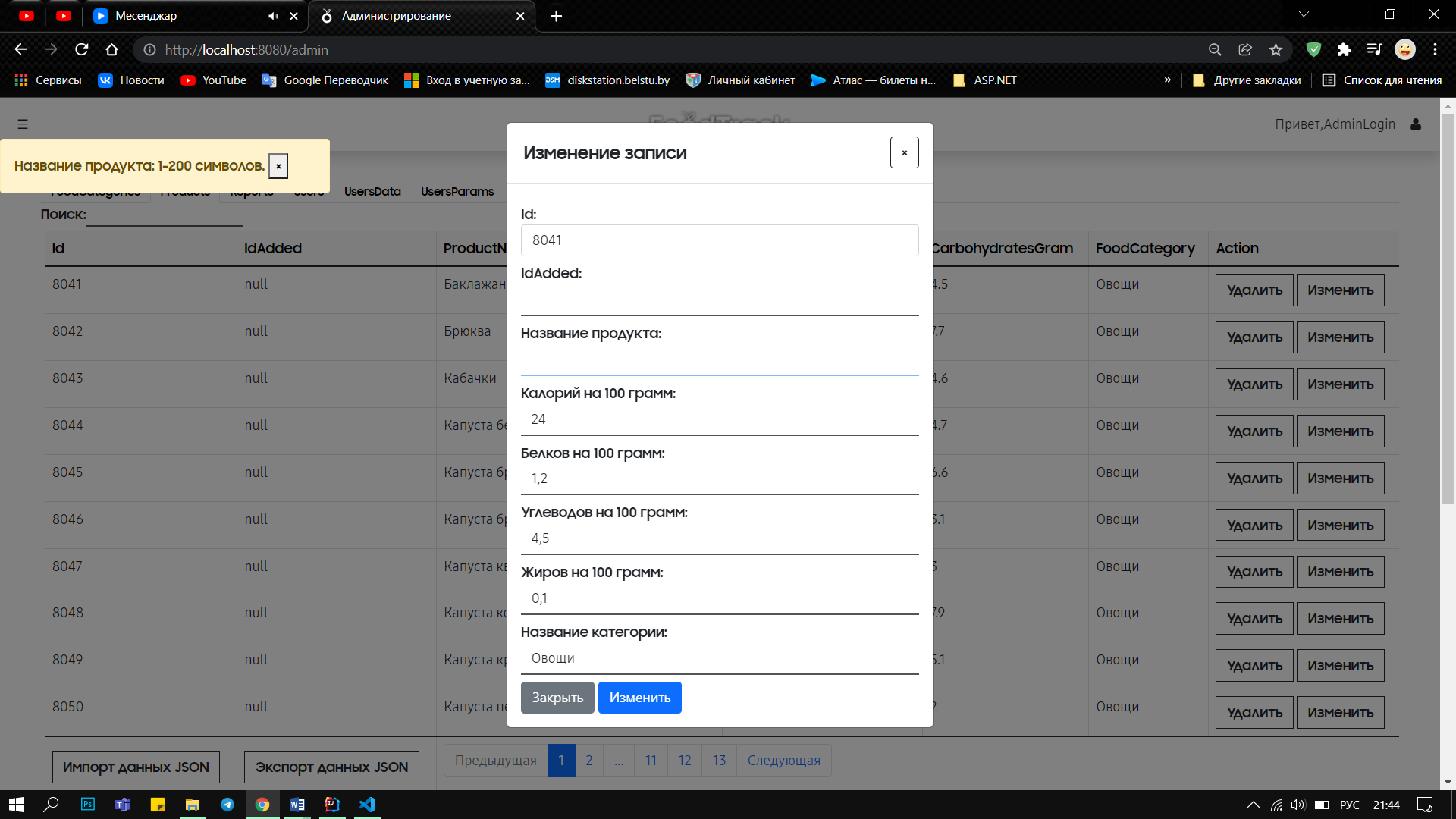


Рисунок 4.11 – Уведомление о вводе некорректных данных

После успешного удаления записи пользователю выводится уведомление. Пример уведомления изображён на рисунке 4.12.

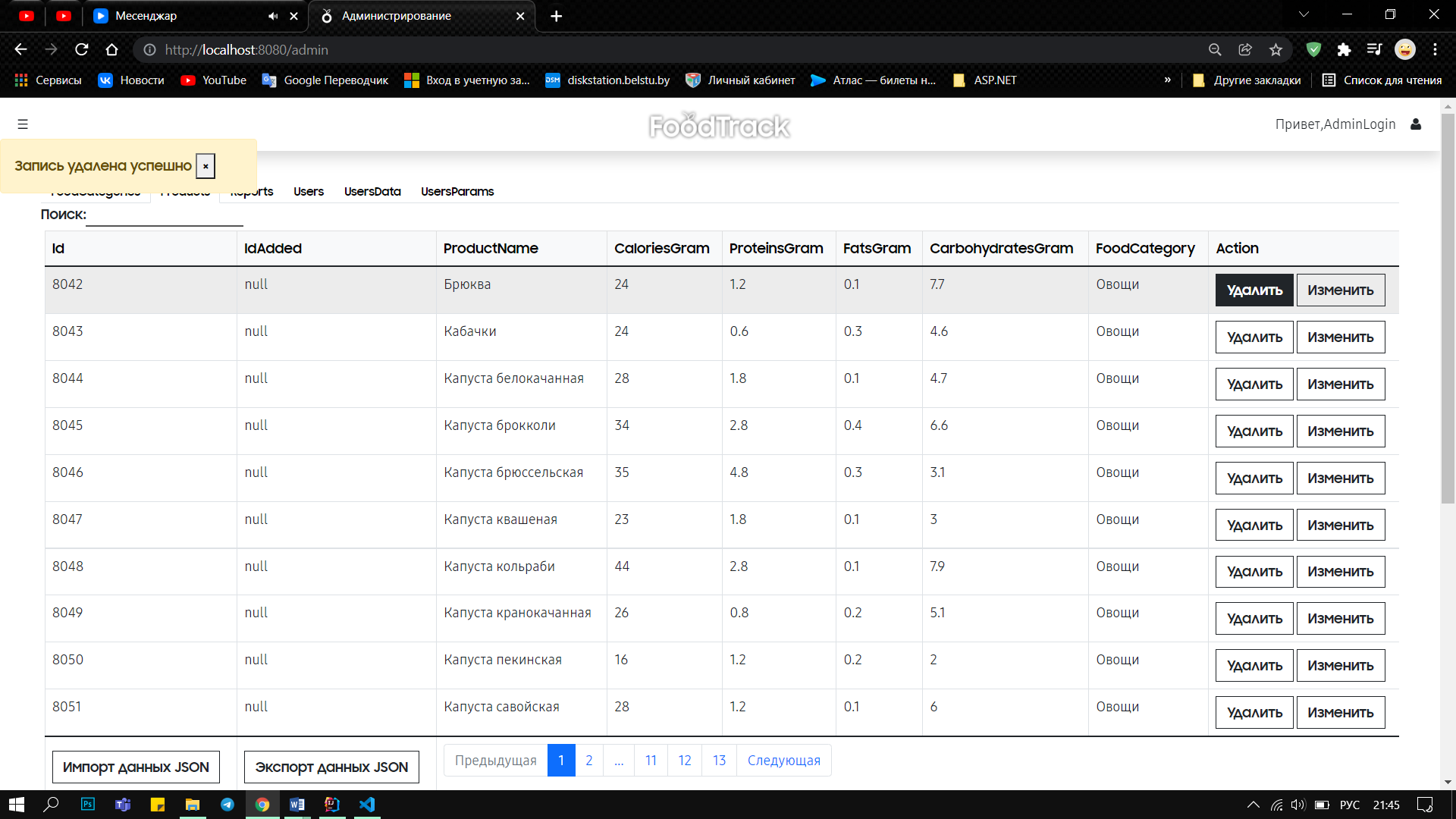


Рисунок 4.12 – Уведомление об успешном удалении записи

При нажатии на кнопку импорта данных в формате JSON открывается диалоговое окно, предлагающее пользователю выбрать файл, содержащий JSON строку, для вставки значений в таблицы Products и FoodCategories. После завершения процесса внесения данных в таблицу пользователь получается сообщение о количестве удачно вставленных строк в таблицы. Пример отображаемого уведомления изображён на рисунке 4.13.

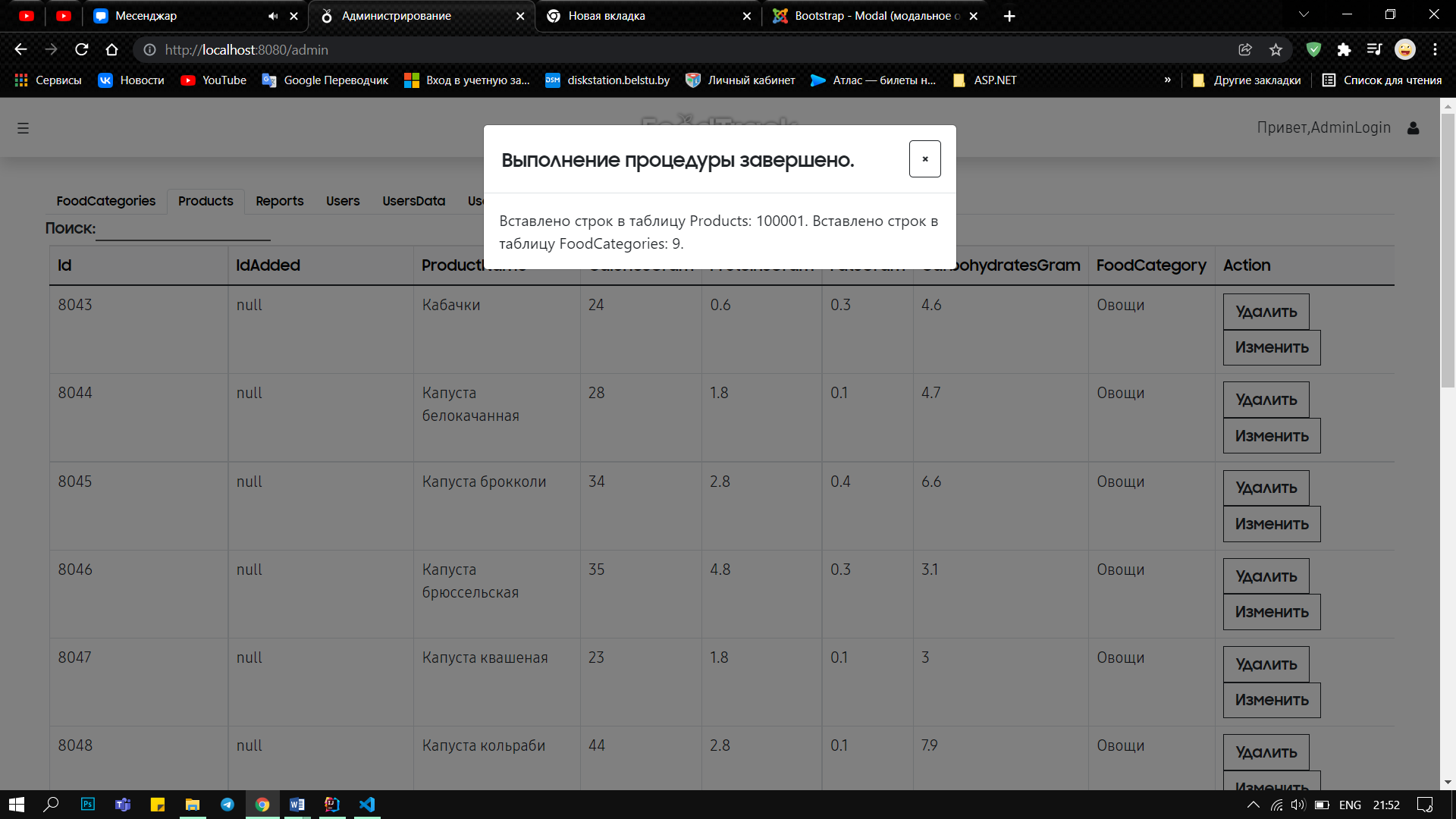


Рисунок 4.13 – Уведомление об успешном добавлении данных

Каждая из вкладок навигационного меню содержит поле поиска, текстовый запрос из которого берётся для выборки данных из таблицы по сравнению значения каждого из полей с текстом запроса. Пример результата поискового запроса изображён на рисунке 4.14.



Рисунок 4.14 – Результат поискового запроса

В левом верхнем углу каждой страницы приложения располагается кнопка, предоставляющая пользователю просмотреть навигационное меню всего приложения. Содержимое меню изображено на рисунке 4.15.

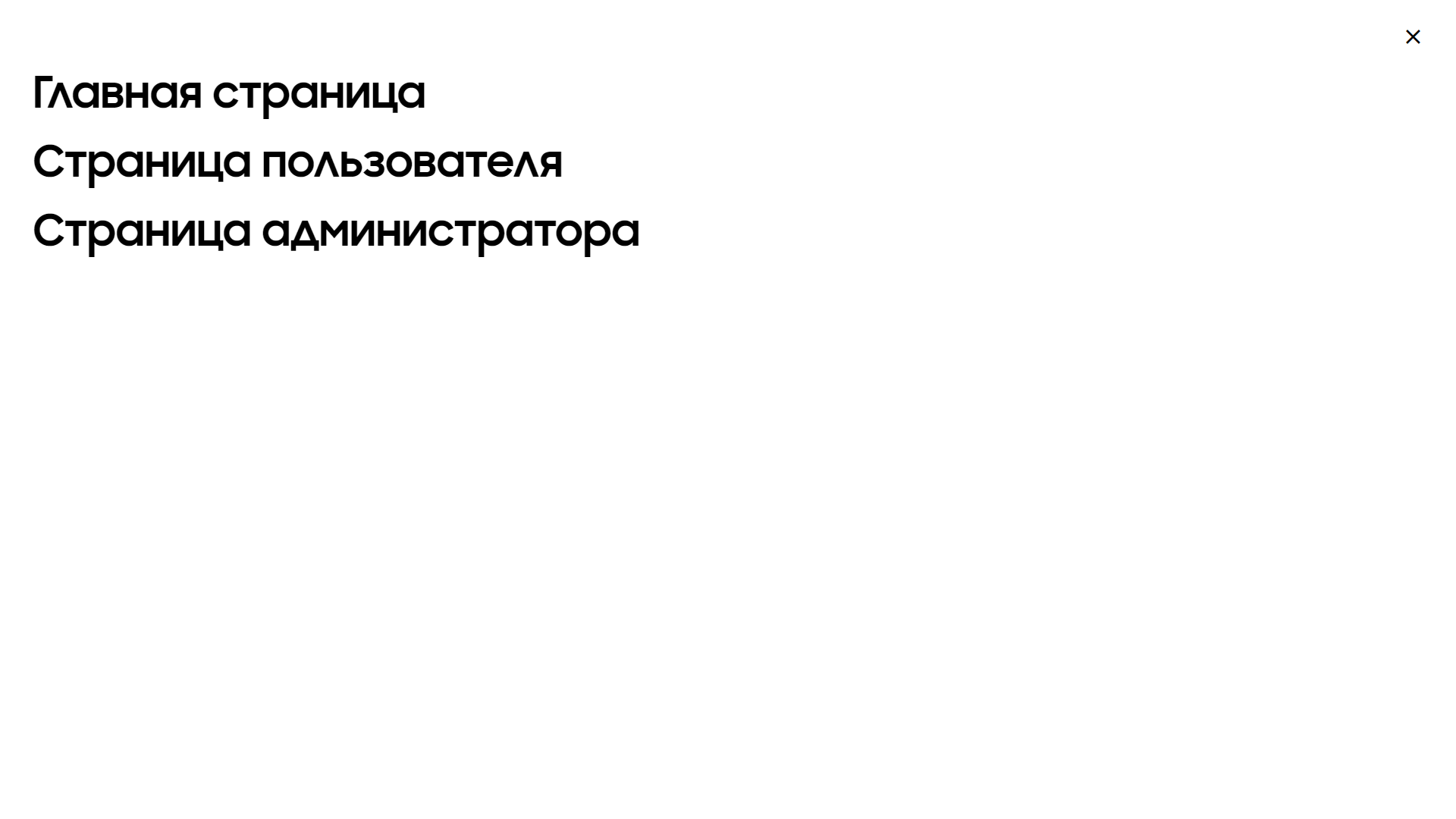


Рисунок 4.15 – Навигационное меню приложения

На странице регистрации расположена форма для ввода данных, которые после успешного завершения процесса регистрации будут использованы для формирования первого отчёта о физических параметрах пользователя, записи для информации о пользователе и его аккаунте. Страницы представлена на рисунке 4.16.

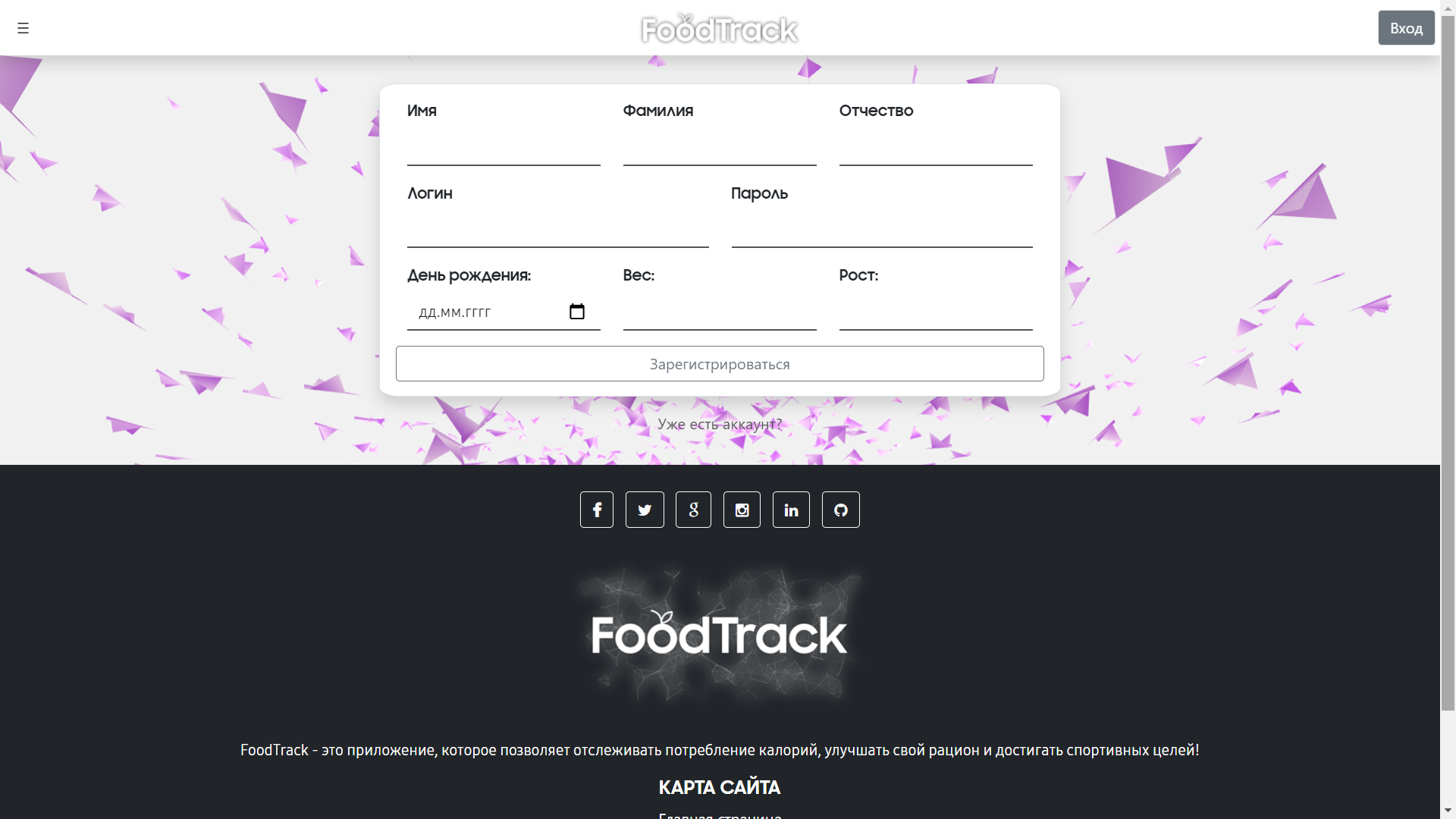


Рисунок 4.16 – Страница регистрации

После завершения процесса регистрации пользователь будет перенаправлен на страницу авторизации.

Заключение

В процессе решения поставленной задачи была достигнута поставленная цель по созданию базы данных «Контроль пищевых привычек», взаимодействующая с приложением для управления действиями пользователя и расширения набора данных, используемого пользователем во время работы с приложением. Основой целью курсового проекта стало проектирование базы данных для дальнейшей интеграции с приложением, которое помогло облегчить взаимодействие с базой данных посредством программного интерфейса. При разработке выполнены следующие пункты:

* Регистрация и авторизация пользователей и администраторов приложения;
* Функционал приложения разделён на сервер и клиент;
* Для созданной базы данных была создана многомерная структура данных;
* Предоставлена возможность поиска данных по таблицам;
* Изменение, удаление и добавление данных во все таблицы базы данных;
* Контроль действий пользователя через возможность изменения их данных и предоставления привилегий администратора;
* Импорт и экспорт данных в формате JSON.

В курсовом проекте были реализованы следующие задачи:

* Создание базы данных;
* Создание сервера;
* Создание клиента с пользовательским интерфейсом;
* Реализация функций работы приложения;
* Тестирование программного продукта.

Данный проект является примером результата разработки базы данных и приложения для работы с ней. Приложение предоставляет достаточный функционал для работы с данными и позволяет пользователю полноценно контролировать свой рацион.

# Список литературы

1. Основные характеристики olap систем [Электронный ресурс] / Центр развития компетенций в бизнес-информатике Высшей школы бизнеса. – Режим доступа: https://hsbi.hse.ru/articles/osnovnye-kharakteristiki-olap-sistem/. – Дата доступа: 15.10.2021.

2. Guides [Электронный ресурс] / VMware, Inc. or its affiliates – Режим доступа: https://spring.io/guides. – Дата доступа: 01.11.2021.

3. Spring MVC: создание веб-сайтов и RESTful сервисов [Электронный ресурс] / Habr. – Режим доступа: https://habr.com/ru/post/500572/. – Дата доступа: 12.11.2021.

4. Доступ к данным через JPA [Электронный ресурс] / Spring.io. – Режим доступа: http://spring-projects.ru/guides/accessing-data-jpa/. – Дата доступа: 15.11.2019.

5. Постраничная навигация по товарам в интернет-магазине [Электронный ресурс] / Webdevkin. – Режим доступа: https://webdevkin.ru/posts/frontend/postranichnaya-navigacziya. – Дата доступа: 03.12.2019.

Приложение А

CREATE TABLE Users

(

Id int constraint PK\_USERS primary key (Id) identity(1,1),

IsAdmin bit NOT NULL default '0',

UserLogin varchar(25) NOT NULL UNIQUE,

UserPassword varchar(60) NOT NULL UNIQUE

)

CREATE TABLE UsersParams

(

Id int constraint PK\_USERS\_PARAMS primary key(Id) identity(1,1),

IdParams int NOT NULL constraint FK\_USERS\_PARAMS\_USERS foreign key (IdParams) references Users(Id) ON DELETE CASCADE,

ParamsDate date NOT NULL,

UserWeight decimal(4,1) default '0' NOT NULL,

UserHeight int default '0' NOT NULL

)

CREATE TABLE UsersData

(

Id int constraint PK\_USERS\_DATA primary key(Id) identity(1,1),

IdData int UNIQUE NOT NULL constraint FK\_USERS\_DATA\_USERS foreign key (IdData) references Users(Id) ON DELETE CASCADE ,

FullName varchar(300) NOT NULL,

Birthday date NOT NULL

)

CREATE TABLE FoodCategories

(

Id int constraint PK\_FCATEGORIES primary key (Id) identity(1,1),

CategoryName varchar(50) NOT NULL UNIQUE

)

CREATE TABLE Products

(

Id int constraint PK\_PRODUCTS primary key(Id) identity(1,1),

IdAdded int constraint FK\_PRODUCTS\_USERS foreign key(IdAdded) references Users(Id) ON DELETE SET NULL,

ProductName varchar(200) NOT NULL UNIQUE,

CaloriesGram decimal(7,2) NOT NULL default '0',

ProteinsGram decimal(7,2) NOT NULL default '0',

FatsGram decimal(7,2) NOT NULL default '0',

CarbohydratesGram decimal(7,2) NOT NULL default '0',

FoodCategory varchar(50) NOT NULL constraint FK\_PRODUCTS\_FCATEGORY foreign key (FoodCategory) references FoodCategories(CategoryName) ON UPDATE CASCADE ON DELETE CASCADE

)

CREATE TABLE Reports

(

Id int constraint PK\_REPORTS primary key(Id) identity(1,1),

IdReport int NOT NULL constraint FK\_REPORTS\_USERS foreign key (IdReport) references Users(Id),

ProductName varchar(200) NOT NULL constraint FK\_REPORTS\_PRODUCTS foreign key (ProductName) references Products(ProductName) ON UPDATE CASCADE ON DELETE CASCADE,

ReportDate date NOT NULL,

EatPeriod varchar(8) NOT NULL,

DayGram decimal(8,2) NOT NULL default '0',

DayCalories decimal(8,2) NOT NULL default '0',

DayProteins decimal(8,2) NOT NULL default '0',

DayFats decimal(8,2) NOT NULL default '0',

DayCarbohydrates decimal(8,2) NOT NULL default '0'

)

Приложение Б

# 