МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных Технологий

Кафедра Информационных систем и технологий

Специальность 1-40 01 01 «Программное обеспечение информационных

технологий»

Специализация Программирование интернет-приложений

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**К КУРСОВОМУ ПРОЕКТУ НА ТЕМУ:**

«Применение технологии SQL Server Reporting Services (SSRS) для задачи «Дневник калорий»

Выполнил студент Пахолко Алена Степановна

(Ф.И.О.)

Руководитель проекта ст.пр. Блинова Е.А.

(учен. степень, звание, должность, подпись, Ф.И.О.)

Курсовой проект защищен с оценкой

Минск 2018

**Содержание**

[Введение 3](#_Toc532983115)

[1. Постановка задачи 4](#_Toc532983116)

[1.1 Обзор и анализ прототипов 4](#_Toc532983117)

[1.2 Актуальность решаемой задачи 6](#_Toc532983118)

[2. Разработка базы данных 7](#_Toc532983119)

[2.1 Проектирование базы данных 7](#_Toc532983120)

[2.2 Процедуры для решения поставленных задач 8](#_Toc532983121)

[2.2.1 Процедуры получения данных из таблиц 8](#_Toc532983122)

[2.2.2 Процедуры сбора данных для отчетов 9](#_Toc532983123)

[2.2.3 Процедуры удаления данных 9](#_Toc532983124)

[2.2.4 Процедуры чтения данных из XML 10](#_Toc532983125)

[2.2.5 Процедуры записи данных в XML 10](#_Toc532983126)

[2.3 Триггеры 11](#_Toc532983127)

[2.4 Индексы 11](#_Toc532983128)

[2.5 Представления 11](#_Toc532983129)

[2.6 Привилегии 12](#_Toc532983130)

[2.7 Функции 12](#_Toc532983131)

[3. Технология SQL Server Reporting Services (SSRS) 13](#_Toc532983132)

[4. Руководство пользователя 19](#_Toc532983133)

[5. Тестирование 25](#_Toc532983134)

[Заключение 28](#_Toc532983135)

[Список использованных источников 29](#_Toc532983136)

[Приложение А 30](#_Toc532983137)

[Приложение Б 31](#_Toc532983138)

# Введение

Научно-техническая революция к концу 20 века почти полностью освободила человека от тяжелого физического труда, сократив его энерготраты примерно на 1000 ккал/сутки. Такие изменения должны были привести к существенному уменьшению объема потребляемой пищи. Однако если эти уменьшившиеся объемы могут позволить человеку удовлетворить потребности в основных пищевых веществах - белках, жирах, углеводах и энергии, то они не способны обеспечить поступление витаминов, минеральных веществ и других биологически активных компонентов пищи в необходимых количествах.

Образовавшиеся «ножницы» и являются той объективной причиной, по которой современный человек не может составить адекватный по энергетической ценности рацион из обычных натуральных продуктов, и в результате он или переедает, набирая лишний вес, либо недополучает биологически значимые пищевые компоненты, дефицит которых наблюдается в питании населения большинства экономически развитых государств. Более того, мировое медицинское сообщество оценивает сложившуюся ситуацию в питании как кризисную.

Помимо ожирения, которое стремительно распространяется среди всех слоев населения, стоит отметить еще один факт, связанный с проблемами образования в области культуры питания и касающийся особенностей питания достаточно большой группы молодежи. Не обладая знаниями о реальных потребностях своего организма в пищевых и биологически активных веществах, руководствуясь только стремлением соответствовать модным стандартам внешности, молодые люди пренебрегают правилами здорового питания, что зачастую приводит к возникновению очень серьезного и тяжелого заболевания – анорексии.

Мода на здоровый образ жизни не прошла мимо индустрии высоких технологий. В современном мире есть большое количество приложений, которые помогут вести по-настоящему здоровый образ жизни посредством разнообразных фитнес-браслетов, «умных» часов и обычных смартфонов. Эти приложения позволяют контролировать физическую активность, правильное питание, количество и качество сна, помогают правильно проводить тренировки.

# 1.Постановка задачи

## 1.1 Обзор и анализ прототипов

На сегодняшний день существует большое количество программных средств, помогающих людям организовывать свой ежедневный рацион, следить за нормами потребления калорий, белков, жиров и углеводов, распределять физическую нагрузку и т.д.

На самых популярных мобильных платформах iOS и Android есть два «родных» приложения — «Здоровье» у Apple и Google Fit у Google, — которые могут собирать информацию из сторонних приложений, а также позволяют им обмениваться этой информацией между собой. Это идеальная опция для перфекционистов, так как она позволяет одновременно пользоваться несколькими приложениями, беря из каждого самое лучшее. Например, из одного — продвинутую систему тренировок, из другого — возможность отслеживать любые виды физической активности, из третьего — огромную базу продуктов питания и разные системы правильного питания.

**MyFitnessPal**

MyFitnessPal — одно из самых многофункциональных приложений для поддержания здорового образа жизни. В нем есть простой и удобный счетчик калорий с большой базой данных о продуктах, содержащей более 4 млн наименований. Приложение в состоянии отслеживать пройденное расстояние, также для подсчета сожженных калорий в него можно загружать спортивные и фитнес-тренировки. При желании можно соблюдать диету в компании с друзьями, добавлять подруг, друзей, делиться достижениями и поддерживать друг друга. На рисунке 1.1 приведен пример данного приложения.

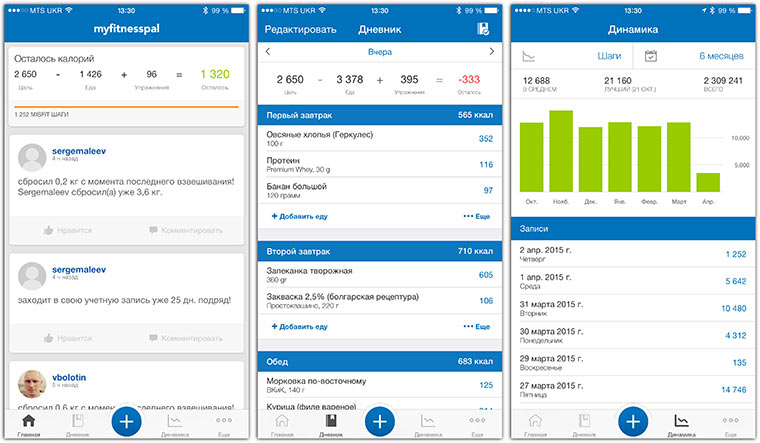


Рисунок 1.1 - MyFitnessPal

**FatSecret**

Счетчик Калорий от FatSecret – еще одно удобное приложение, которое позволяет не только вести подсчет калорий, но и записывать все данные во встроенный дневник веса и собирать при этом необходимую информацию о пройденных тренировках. Среди плюсов программы можно выделить считывание штрих-кодов, дневник веса и подробная статистика о результатах. На рисунке 1.2 приведен пример данного приложения.

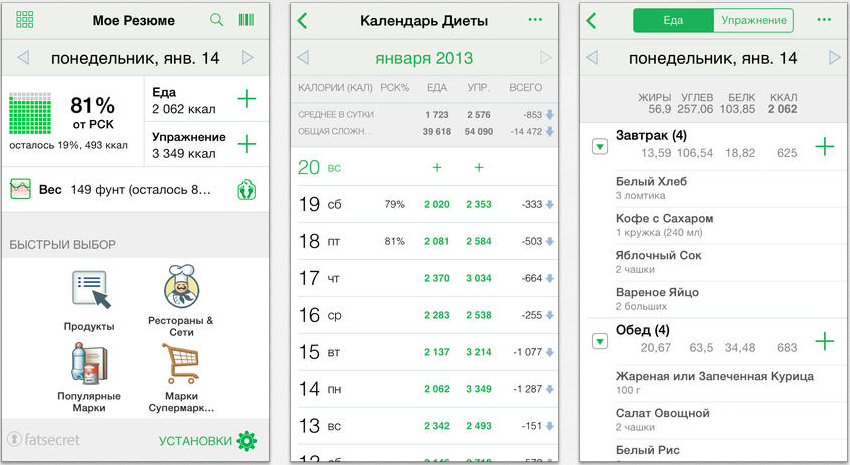


Рисунок 1.2 - FatSecret

**Dia-Life**

Если говорить коротко, то это сборник онлайн инструментов для анализа и планирования питания, в котором также присутствуют разделы для людей, живущих с сахарным диабетом. Что в нем есть: калькулятор калорий, большая база продуктов с подробным описанием их составов, контроль физической активности, дневник для отслеживания прогресса, рационы, блюда и диеты.

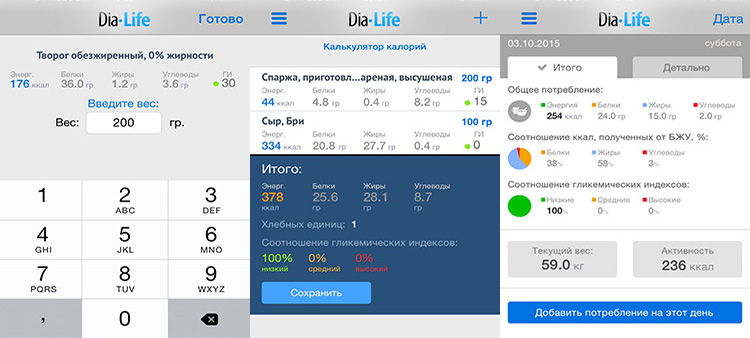


Рисунок 1.3 – Dia-Life

Если подвести итоги и выделить некоторую общую функциональность для всех приложений данной тематики, то она будет в себя включать следующее:

* калькулятор калорий;
* большая база продуктов;
* контроль потребления пищи;
* отслеживание прогресса.

Учитывая данную информацию и была спроектирована база данных для решения проблемы управления режимом питания и разработано приложение для работы с ней.

## 1.2 Актуальность решаемой задачи

Питание является одним из важнейших факторов, оказывающих влияние на здоровье, работоспособность и устойчивость организма человека к воздействию экологически вредных факторов. Вопросы питания населения в настоящее время являются крупной физиолого-гигиенической проблемой. Несбалансированность структуры продуктовых наборов и фактического питания сопровождается нарушениями физического развития, напряжённостью обменных процессов и адаптационных механизмов, высоким уровнем заболеваемости.

Современный человек потребляет недостаточное количество необходимых веществ в результате монотонности рациона, сведения к узкому стандартному набору нескольких основных групп продуктов и готовых блюд. В следствии чего возникает необходимость контроля питания человека. Важно подчеркнуть, что норма питания для каждого человека не является некой постоянной величиной. Напротив, эта величина переменная, она изменяется с возрастом, зависит от пола, роста, веса человека, уровня физической и психоэмоциональной активности, состояния здоровья и действия внешних факторов.

# 2. Разработка базы данных

## 2.1 Проектирование базы данных

Для хранения и предоставления данных пользователю, была разработана база данных, которая содержит в себе семь таблиц, все связаны друг с другом внешними ключами. Диаграмма представлена на рисунке 2.1.

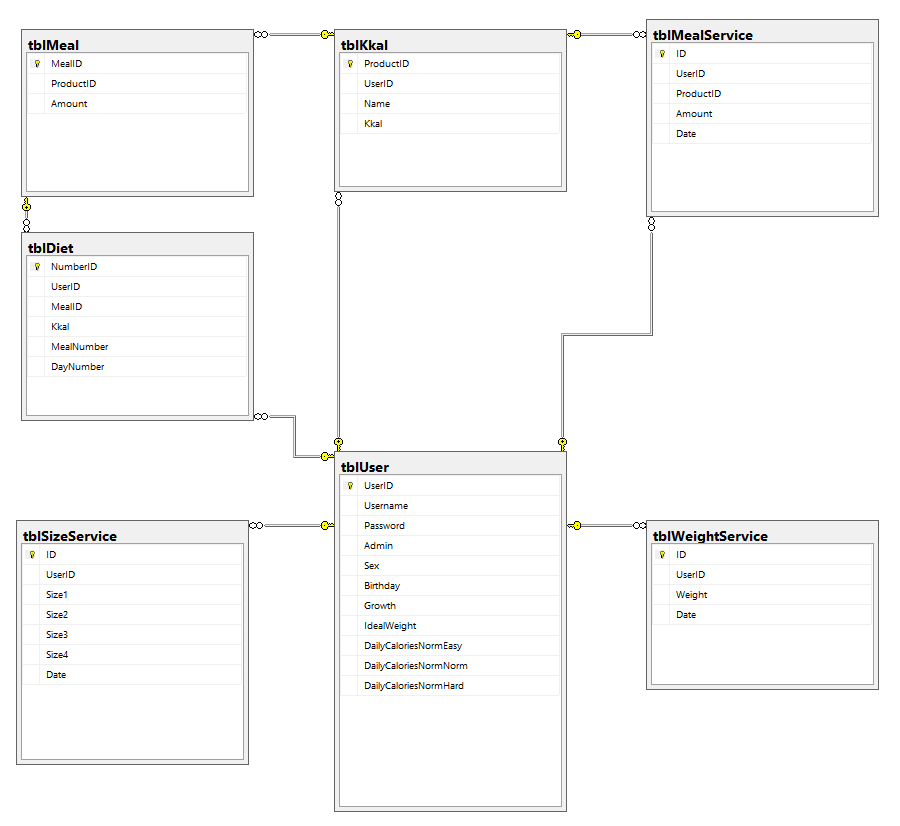


Рисунок 2.1 – Диаграмма базы данных

Таблица tblUser содержит всю информацию о пользователях приложения, а именно: идентификатор пользователя (UserID int PK), логин (Username nvarchar(50)), пароль (Password nvarchar(50)), роль (Admin int), пол (Sex nvarchar(6)), возраст (Birthday int), рост (Growth decimal), идеальный вес (IdealWeight decimal), три режима потребления калорий (DailyCaloriesNormEasy decimal, DailyCaloriesNormNorm decimal, DailyCaloriesNormHard decimal).

Таблица tblKKal содержит информацию о всех продуктах в приложении, а именно: идентификатор продукта (ProductID int PK), область видимости (UserID int FK), название продукта (Name nvarchar(50)), калорийность (Kkal int).

Таблица tblMeal содержит информацию о порциях продукта, а именно: идентификатор порции (MealID int PK), идентификатор продукта (ProductID int FK), количество продукта в граммах (Amount decimal).

Таблица tblDiet содержит информацию о составленном рационе пользователя, а именно: идентификатор приема пищи (NumberID int PK), идентификатор пользователя (UserID int FK), идентификатор порции (MealID int FK), калорийность дня (поле, нарушающее нормализацию, Kkal int), порядковый номер приема пищи (MealNumber int), порядковый номер дня приема пищи (DayNumber int).

Таблица tblMealService содержит информацию о каждом приеме пищи пользователей, а именно: идентификатор (ID int PK), идентификатор пользователя (UserID int FK), идентификатор продукта (ProductID int FK), количество в граммах (Amount decimal), дата (Date date).

Таблица tblWeightService содержит информацию об измерениях веса пользователей, а именно: идентификатор (ID int PK), идентификатор пользователя (UserID int FK), результаты измерения (Weight decimal), дата (Date date).

Таблица tblSizeService содержит информацию об измерениях охватов пользователей, а именно: идентификатор (ID int PK), идентификатор пользователя (UserID int FK), шея (Size1 decimal), грудь (Size2 decimal), талия (Size3 decimal), бедра (Size4 decimal), дата (Date date).

Таким образом, для ведения истории пользователя используются таблицы tblMealService (употребление пищи), tblWeightService (изменения веса), tblSizeService (изменения обхватов). Для составления индивидуального рациона пользователя используются таблицы tblMeal и tblDiet.

## 2.2 Процедуры для решения поставленных задач

### 2.2.1 Процедуры получения данных из таблиц

Разработаны и созданы процедуры для получения данных из различных таблиц. Далее приведен пример процедуры получения пользователя по имени и паролю.

Листинг:

--SELECT USER

GO

ALTER procedure [dbo].[SelectUser]

(

@username nvarchar(50), --логин

@password nvarchar(50) --пароль

)

as begin

begin try

select \* from [dbo].[tblUser] --выборка

where [Username]=@username and [Password]=@password;

end try

begin catch

print error\_message()

end catch;

end;

### 2.2.2 Процедуры сбора данных для отчетов

В качестве примера для данных процедур представлена процедура сбора данных для отчета по дневному потреблению калорий. В теле процедуры используется функция SelectOstKkalUser, возвращающая скалярное значение.

Листинг:

GO

ALTER procedure [dbo].[ReportDayAlreadyEatKkal]

(

@userid int

)

as begin

begin try

declare @kkal decimal(18,0) =(select [dbo].[SelectOstKkalUser](@userid)); --остаток калорий

declare @kkalNorm decimal(18,0) = (select [DailyCaloriesNormNorm]

from [dbo].[tblUser] where [UserID]=@userid); --норма калорий на день

declare @del decimal(18,0) = (@kkal/@kkalNorm)\*100; --процентная часть потребленных калорий

if (@del >100) set @del = 100; --если потребили больше нормы

select @userid as [UserIDReport],GetDate() as [DateReport],@kkal as [KkalReport],@kkalNorm as [KkalNorm],@del as [Result] ;

end try

begin catch

print error\_message()

end catch

end; -- Конец процедуры

### 2.2.3 Процедуры удаления данных

В качестве примера представлена процедура удаления данных о всех порциях того или иного продукта, добавленных определенным пользователем. В качестве параметра указывается идентификатор пользователя. В теле процедуры используется проверка по продуктам самого пользователя.

Листинг:

GO

ALTER procedure [dbo].[DeleteMeal]

(

@userid int

)

as begin

begin try

begin tran

delete from [dbo].[tblMeal] where [ProductID] --удаление по связи с табл продуктов

in (select [ProductID] from [dbo].[tblKkal] where [UserID] = @userid);

commit;

end try

begin catch

print error\_message()

end catch;

end;

### 2.2.4 Процедуры чтения данных из XML

В качестве примера для данных процедур представлена процедура для добавления данных о всех протоколах изменения обхватов пользователей, зарегистрированных в приложении, из xml-файла в базу данных.

Листинг:

GO

ALTER PROCEDURE [dbo].[ImportS]

@xml XML = NULL

AS

Select @xml =

CONVERT(XML,bulkcolumn,2) FROM OPENROWSET(BULK 'D:\Study\3k1s\Kursach\PipDiet\DB\1Size.xml',SINGLE\_BLOB) AS X

SET ARITHABORT ON

Insert into [dbo].[tblSizeService]

(

[UserID], [Size1], [Size2], [Size3], [Size4], [Date]

)

Select

P.value('UserID[1]', 'int') AS [UserID],

P.value('Size1[1]', 'decimal(18,0)') AS [Weight],

P.value('Size2[1]', 'decimal(18,0)') AS [Weight],

P.value('Size3[1]', 'decimal(18,0)') AS [Weight],

P.value('Size4[1]', 'decimal(18,0)') AS [Weight],

P.value('Date[1]', 'date') AS [Date]

From @xml.nodes('/Sizes/Size') PropertyFeed(P)

### 2.2.5 Процедуры записи данных в XML

В качестве примера для данных процедур представлена процедура для добавления данных о всех протоколах изменения обхватов пользователей, зарегистрированных в приложении, в формат xml.

Листинг:

GO

ALTER PROCEDURE [dbo].[ExportS]

AS

SELECT [UserID], [Size1], [Size2], [Size3], [Size4], [Date]

FROM [dbo].[tblSizeService]

FOR XML PATH('Size'), ROOT('Sizes');

end;

## 2.3 Триггеры

Для работы с протоколами зарегистрированных пользователей необходимо автоматически заносить дату измерений в базу данных. Для выполнения этой задачи были разработаны триггеры, которые заносят текущую дату в поле значения даты измерений. Для примера представлен триггер для вставки даты при добавлении записи о новом взвешивании пользователя.

Листинг:

GO

CREATE TRIGGER [dbo].[WeightS\_INSERT\_UPDATE]

ON [dbo].[tblWeightService]

AFTER INSERT, UPDATE

AS

begin

UPDATE [dbo].[tblWeightService]

SET [Date]=GetDate()

WHERE [ID] = (SELECT [ID] FROM inserted)

end;

## 2.4 Индексы

Для оптимизации работы с поисковыми запросами, необходимо использовать индексы. Так, для работы с наиболее часто используемыми таблицами, были разработаны наборы индексов по наиболее используемым выборкам. Для примера представлен некластеризованный индекс по таблице tblKkal, поля - UserID и Kkal.

Листинг:

create nonclustered index [KkalSost\_index] on [dbo].[tblKkal] (UserID, Kkal);

## 2.5 Представления

Для работы с протоколами зарегистрированных пользователей, составлением отчетов и сопоставлением данных о рационе необходимо использовать данные из разных таблиц, используя выборку по определенному условию. Для выполнения этой задачи были созданы представления. Для примера представлено представление для отображения полной информации о рационе пользователей.

Листинг:

go

alter view [dbo].[DietView]

(UserID,DietID,MealID,DietNum,MealNum,ProdID,NameProd,Amount,Kkal)

as select

a.UserID,a.NumberID,a.MealID,a.DayNumber,a.MealNumber,

b.ProductID,c.Name,b.Amount,a.Kkal

from

[dbo].[tblDiet] a, [dbo].[tblMeal] b, [dbo].[tblKkal] c

where

a.MealID=b.MealID and b.ProductID=c.ProductID;

## 2.6 Привилегии

Для базы данных было разработано два типа пользователей. Один из них разработан для пользования под обычным пользователем, а второй для администраторов. Сперва были разработаны логины для подключения к базе данных.

Листинг:

--connection login to user

CREATE LOGIN [userdb] WITH PASSWORD=N'userdb',

DEFAULT\_DATABASE=[PipDietDB], DEFAULT\_LANGUAGE=[русский],

CHECK\_EXPIRATION=OFF, CHECK\_POLICY=ON

GO

ALTER LOGIN [userdb] ENABLE

GO

--connection login to admin

CREATE LOGIN [admindb] WITH PASSWORD=N'admindb',

DEFAULT\_DATABASE=[PipDietDB], DEFAULT\_LANGUAGE=[русский],

CHECK\_EXPIRATION=OFF, CHECK\_POLICY=ON

GO

ALTER LOGIN [admindb] ENABLE

GO

Далее в рамках нашей базы данных, создаем пользователей, использующих эти логины входа. После чего выдаем гранты для каждого из пользователей. Для обычного пользователя создается роль, после чего ею нагружается пользователь нашей базы данных, отвечающий за подключения обычных пользователей. Для администраторов мы даем грант на всю схему.

Листинг:

--создаем пользователей

CREATE USER [user] FOR LOGIN [userdb];

GO

CREATE USER [admin] FOR LOGIN [admindb];

GO

CREATE ROLE [PipDietDB\_user\_Role];

sp\_addrolemember 'PipDietDB\_user\_Role', 'user';

CREATE ROLE [PipDietDB\_admin\_Role];

GRANT EXECUTE ON SCHEMA::[dbo] TO [admin]

sp\_addrolemember 'PipDietDB\_admin\_Role', 'admin';

end;

## 2.7 Функции

Для работы с расчетами и формулами, а также активностью пользователя в течении дня необходимо использовать набор разработанных функций, возвращающих скалярное значение. Для примера представлена функция для подсчета потребленных калорий за текущий день пользователя.

Листинг:

go

create function [dbo].[SelectOstKkalUser]

(

@userid int

)

returns decimal

as begin

declare @am decimal, --кол-во продукта в граммах

@kkal int, --калорийность продукта

@sum decimal=0, --потребленные за текущий день калории

@prodid int, --идентификатор продукта

@ret decimal = 0, --итоговое значение

@CURSOR CURSOR --курсор для обработки записей протокола текущего дня

SET @CURSOR = CURSOR SCROLL

FOR

SELECT [ProductID],[Amount] FROM [dbo].[tblMealService] WHERE [UserID]= @userid and [Date]=convert(date,GetDate());

OPEN @CURSOR

FETCH NEXT FROM @CURSOR INTO @prodid,@am

WHILE @@FETCH\_STATUS = 0

BEGIN

set @kkal = (select [Kkal] from [dbo].[tblKkal] where [ProductID]=@prodid);

set @sum = @sum + (@kkal\*@am/100); --подсчет потребленных калорий

FETCH NEXT FROM @CURSOR INTO @prodid,@am

END;

CLOSE @CURSOR;

set @ret=@sum;

return @ret;

end;

# 3. Технология SQL Server Reporting Services (SSRS)

SQL Server Reporting Services (SSRS) – это службы для разработки, построения, доставки и просмотра отчетов. С помощью этих служб можно создавать табличные, интерактивные, графические и другие более сложные отчеты с использованием диаграмм и других отчетных элементов.

Данная служба была выбрана в качестве технологии для данной работы по причине того, что разрабатываемая база данных хранит статистические данные об изменении пользователей, их приемах питания, по которым удобнее всего создавать отчеты, чтобы просматривать процессы изменения, статистические данные, информацию о конкретных объектах и ее преобразование и обработку для более приятного и удобного восприятия.

На примере создания отчета, содержащего информацию о потреблении калорий пользователем в течении текущего дня, будет разобрана работа с SSRS.

1. Данный отчет формируется с помощью создания проекта Reporting Project. Для того, чтобы данные из базы данных могли использоваться в представлении отчета, необходимо добавить источник данных (PipDietDB), и затем для каждого отчета определяется свой набор данных (DataSet). Набор данных для данного отчета (рисунок 3.1) формируется на базе процедуры, делающей необходимую выборку по одному параметру – идентификатору пользователя. Данная продедура представлена на рисунке 3.2.

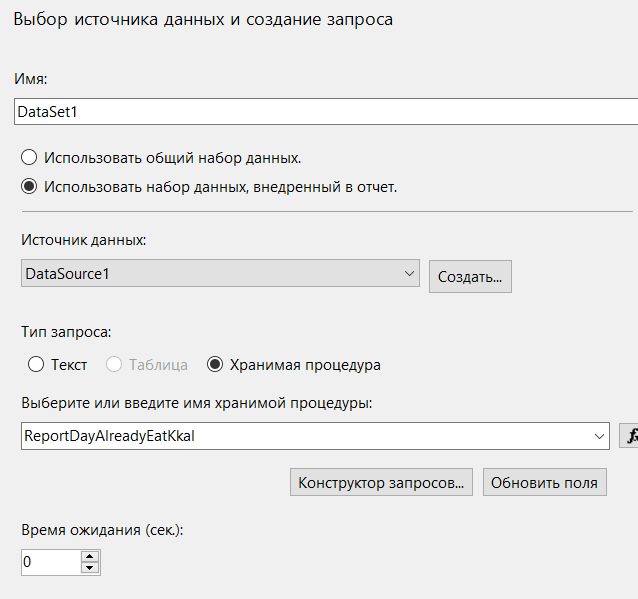


Рисунок 3.1 – Формирование набора данных



Рисунок 3.2 – Процедура получения данных из БД

1. Далее добавим в проект новый элемент – отчет. Определим в его параметрах все необходимые данные (рисунок 3.3).

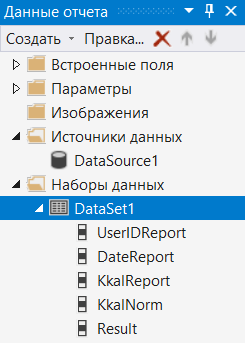


Рисунок 3.3 – Данные отчета

1. В появившемся пустом поле отчета вызовем контекстное меню, и далее выбираем пункт вставить объект. Из всего перечня шаблонов отчетов выбираем Датчик (рисунок 3.4).

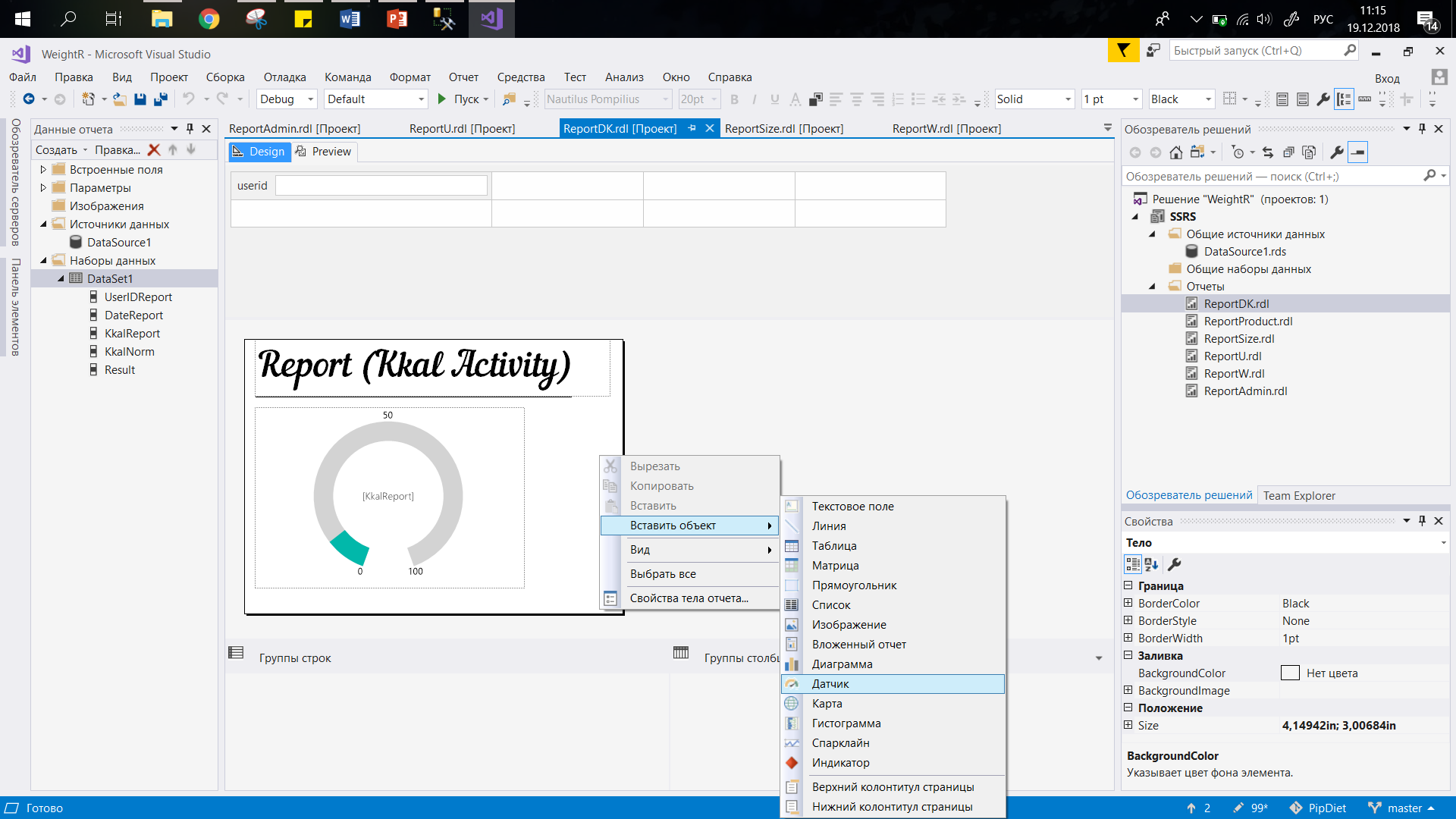


Рисунок 3.4 – Выбор формата отчета

1. Далее следует настройка самого датчика (рисунок 3.5). Так, в поле значения установим поле с суммарным количеством потребленных калорий за текущий день. В поле отображения установим поле, которое отвечает за численное отображение данных калорий. В свойствах самого элемента настраиваем внешний вид (цвет, индикаторы, выражения, интервалы, шкалу и т.д.)

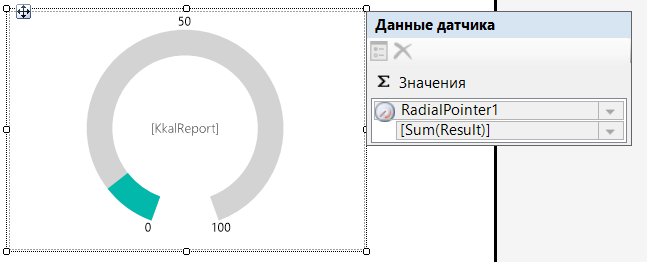


Рисунок 3.5 – Настройка датчика

1. При необходимости добавляем в отчет такие поля, как текстовое поле, линия, область и т.д. Переключаемся во вкладку Preview и вводим необходимые параметры. Наблюдаем формирование отчета. Работа над отчетом завершена (рисунок 3.6).

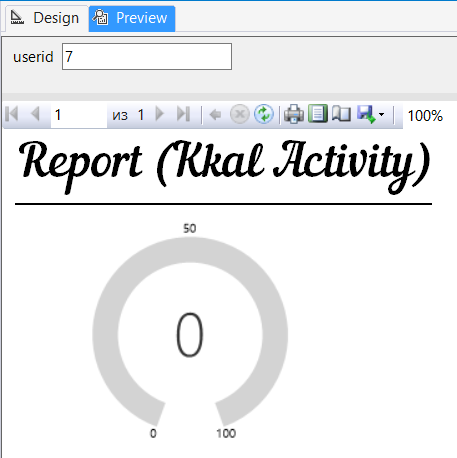


Рисунок 3.6 – Результат

1. В общем итоге в результате работы с технологией SSRS было использованы такие форматы отчетов, как датчик, матрица, список, таблица, текст. Их вид представлен на рисунках



Рисунок 3.7 – Отчет о пользовательских продуктах

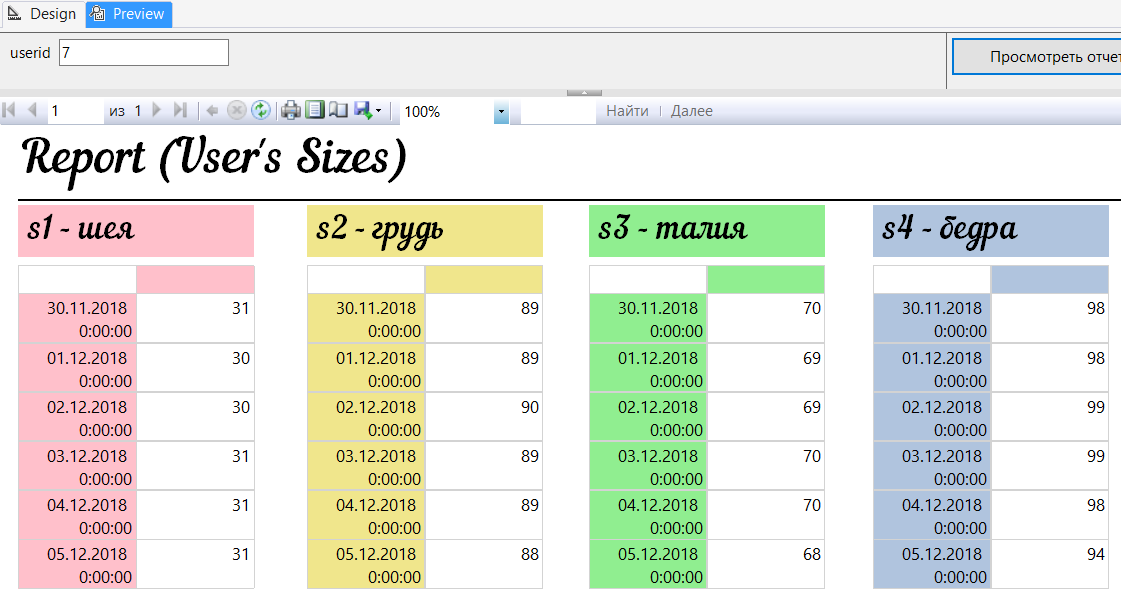


Рисунок 3.8 – Отчет о проколе обхватов пользователя

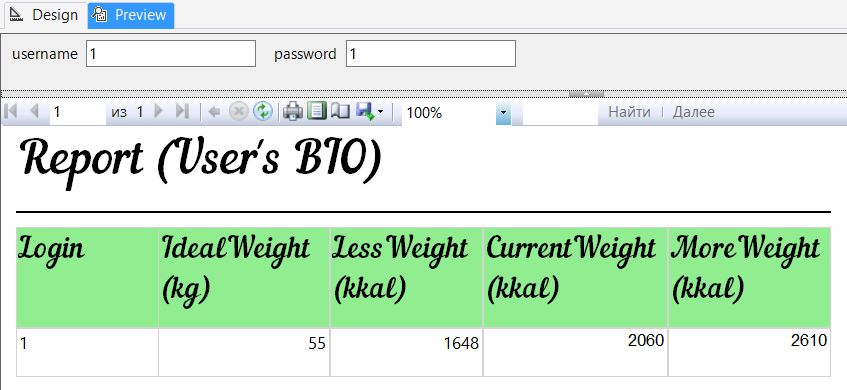


Рисунок 3.9 – Отчет о физиологических данных пользователя

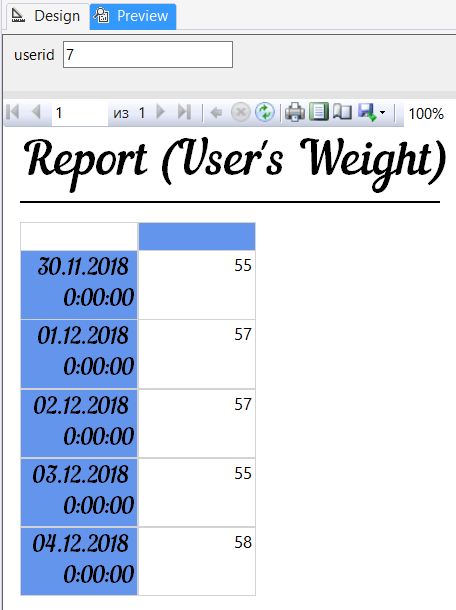


Рисунок 3.10 – Отчет о протоколе веса пользователя

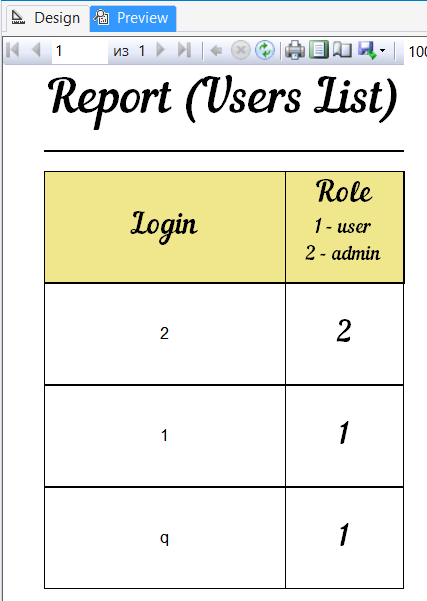
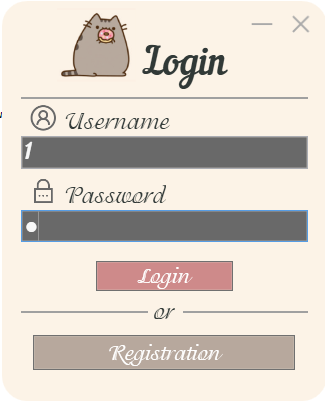
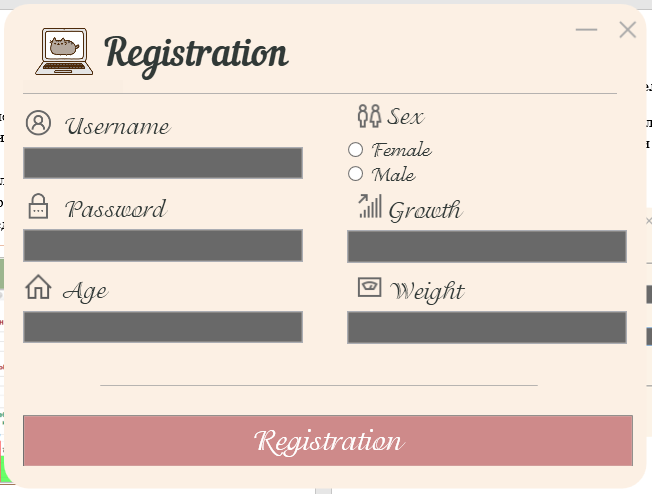


Рисунок 3.11 – Отчет администратора о всех пользователях

# Руководство пользователя

Начальной страницей приложения является вход в приложение. Здесь пользователь может ввести имя и пароль и войти в приложение, либо перейти к регистрации. Данная страница представлена на рисунке 4.1а, а страница регистрации на рисунке 4.1б.

а – Начальная страница б – Страница регистрации

Рисунок 4.1

После входа пользователь попадает на приветственную страницу, где можно увидеть индивидуально рассчитанную статистику для ознакомления. На рисунке 4.2 представлен пример.



Рисунок 4.2 – Страница приветствия

Далее, находясь на любой странице приложения, доступно такое средство навигации как меню. С его помощью можно передвигаться по страницам приложения. Меню для пользователя включает в себя: страница приветствия (рисунок 4.2), страница управления рационом (рисунок 4.3), страница добавления новых данных (рисунки 4.4-4.7), калькулятор калорий (рисунок 4.8), страница настройки (рисунок 4.9). В меню для администратора дополнительно входит окно для управления приложением (рисунок 4.10).

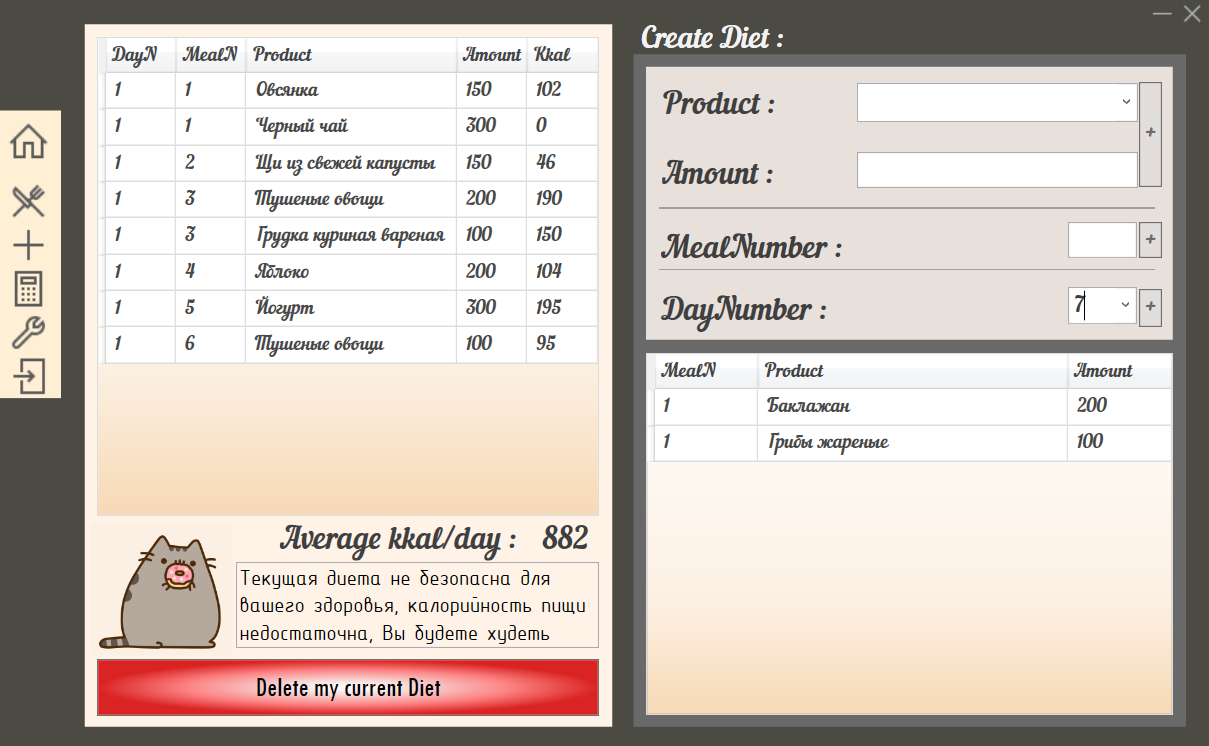


Рисунок 4.3 – Страница управления рационом

На странице управления рационом (рисунок 4.3) происходит последовательная запись данных (вносится набор пищи, задается его порядковый номер, указывается номер дня диеты, заносится в базу данных). На странице также предусмотрена возможность удаления текущей диеты и поле анализа текущего рациона.



Рисунок 4.4 – Добавление данных о приеме пищи



Рисунок 4.5 – Добавление данных о весе



Рисунок 4.6 – Добавление данных об охватах

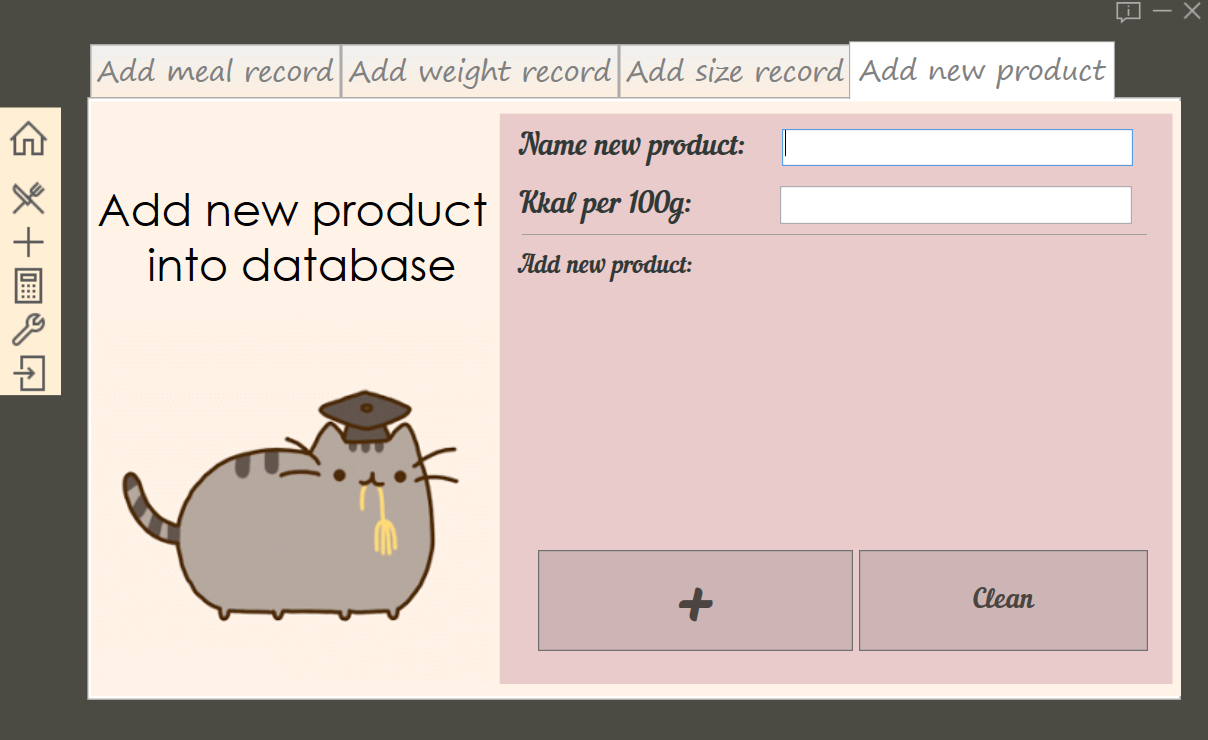


Рисунок 4.7 – Добавление пользовательского продукта в базу данных

На странице добавления новых данных (рисунки 4.4-4.7) вносимые значения используются для ведения протоколов об изменениях параметров пользователей. На вкладке добавления данных о приеме пищи (рисунок 4.4) предусмотрена возможность отображения числа калорий, которые еще можно употребить в течении дня без ущерба здоровью и фигуре.

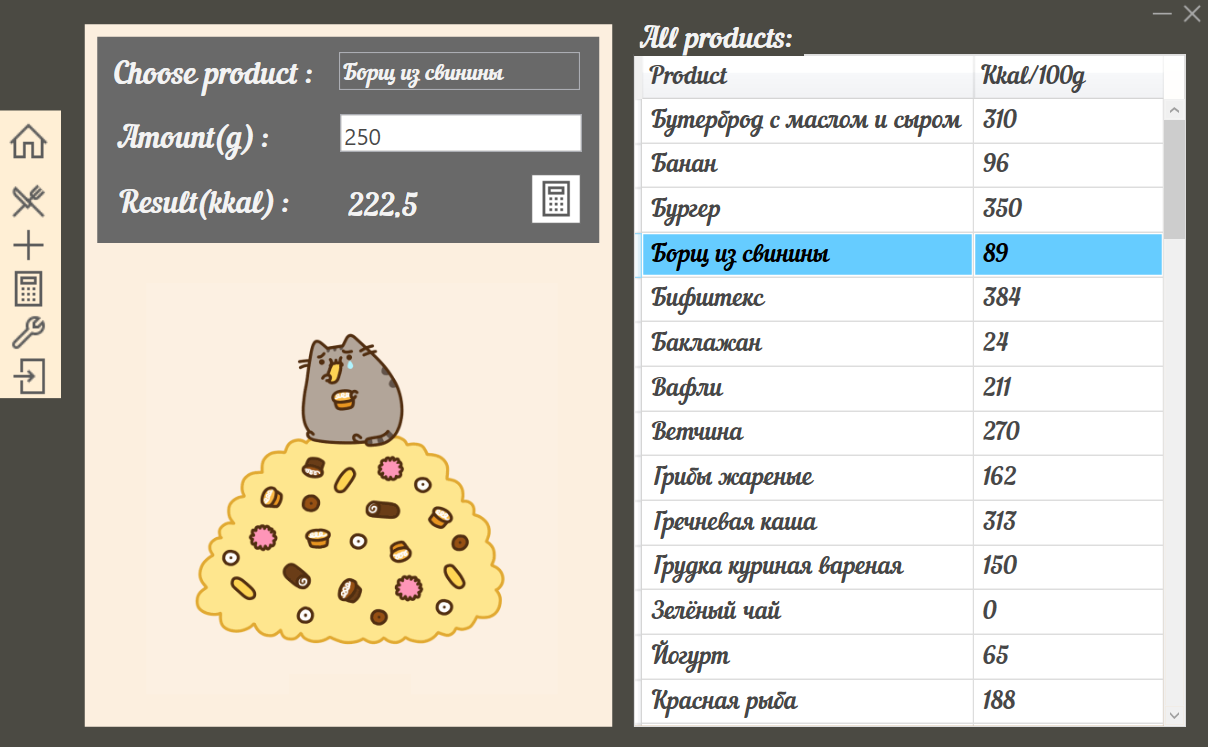


Рисунок 4.8 – Калькулятор калорий

На странице калькулятора калорий (рисунок 4.8) предоставлен список всех доступных данному пользователя продуктов (администраторские и пользовательские). После выбора продукта из списка необходимо указать вес пищи в граммах в соответствующее поле. По нажатию соответствующей кнопки будет отображен результат в калориях.



Рисунок 4.9 – Страница настройки

На странице настройки (рисунок 4.9) доступна возможность смены фона на цвет, картинку, тему по умолчанию.

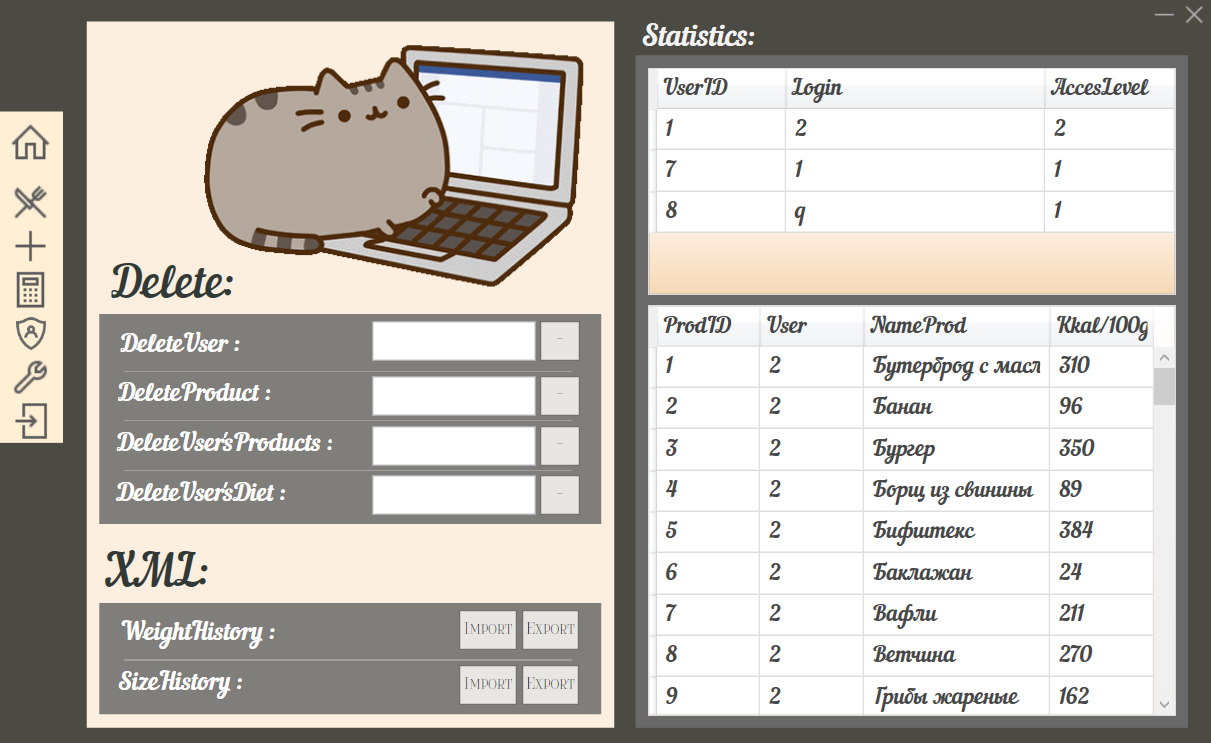


Рисунок 4.10 – Управление приложением

На странице управления приложением отображаются списки всех пользователей приложения, а также список всех продуктов приложения. Выбрав необходимую позицию из списка, можно использовать такой функционал как: удаление пользователя, удаление продукта, удалить все продукты пользователя, удалить рацион пользователя, импорт/экспорт данных протоколов всех пользователей в/из xml-формат.

.

# Тестирование

Для тестирования базы данных были проведены запросы на основе разного объема набора данных, для определения разницы времени в обработке запроса. На объеме данных в 70 строк общий select-запрос на выборку всех данных из таблицы затратил ресурсы, представленные на рисунке 5.1.

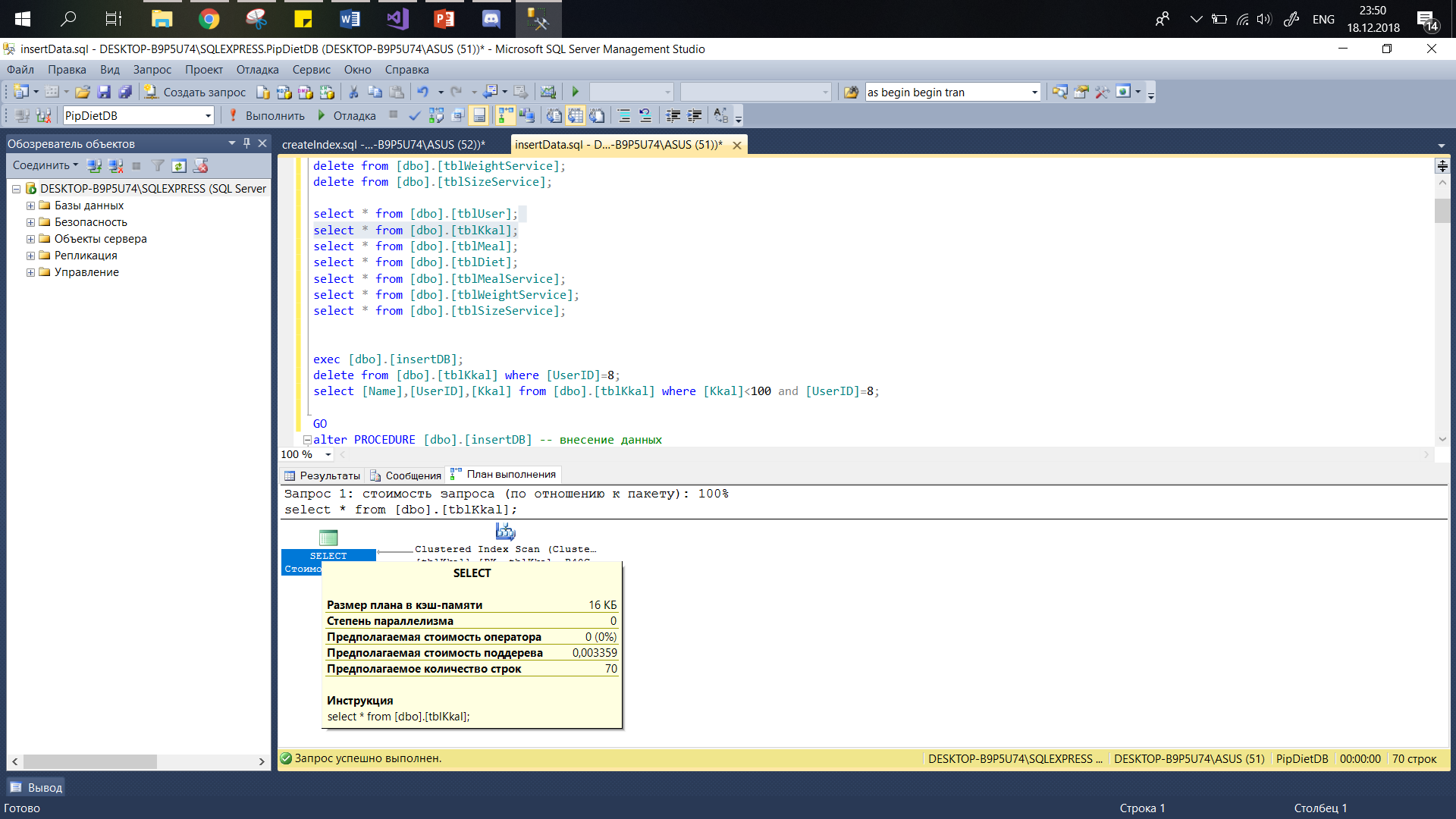


Рисунок 5.1 – Результат select-запроса с 70 строками

После добавления в таблицу Product еще порядка 10 000 данных, результаты уже оказались гораздо ниже, нежели при 70. Результаты такого-же запроса, но к 10 000 данных представлены на рисунке 5.2.

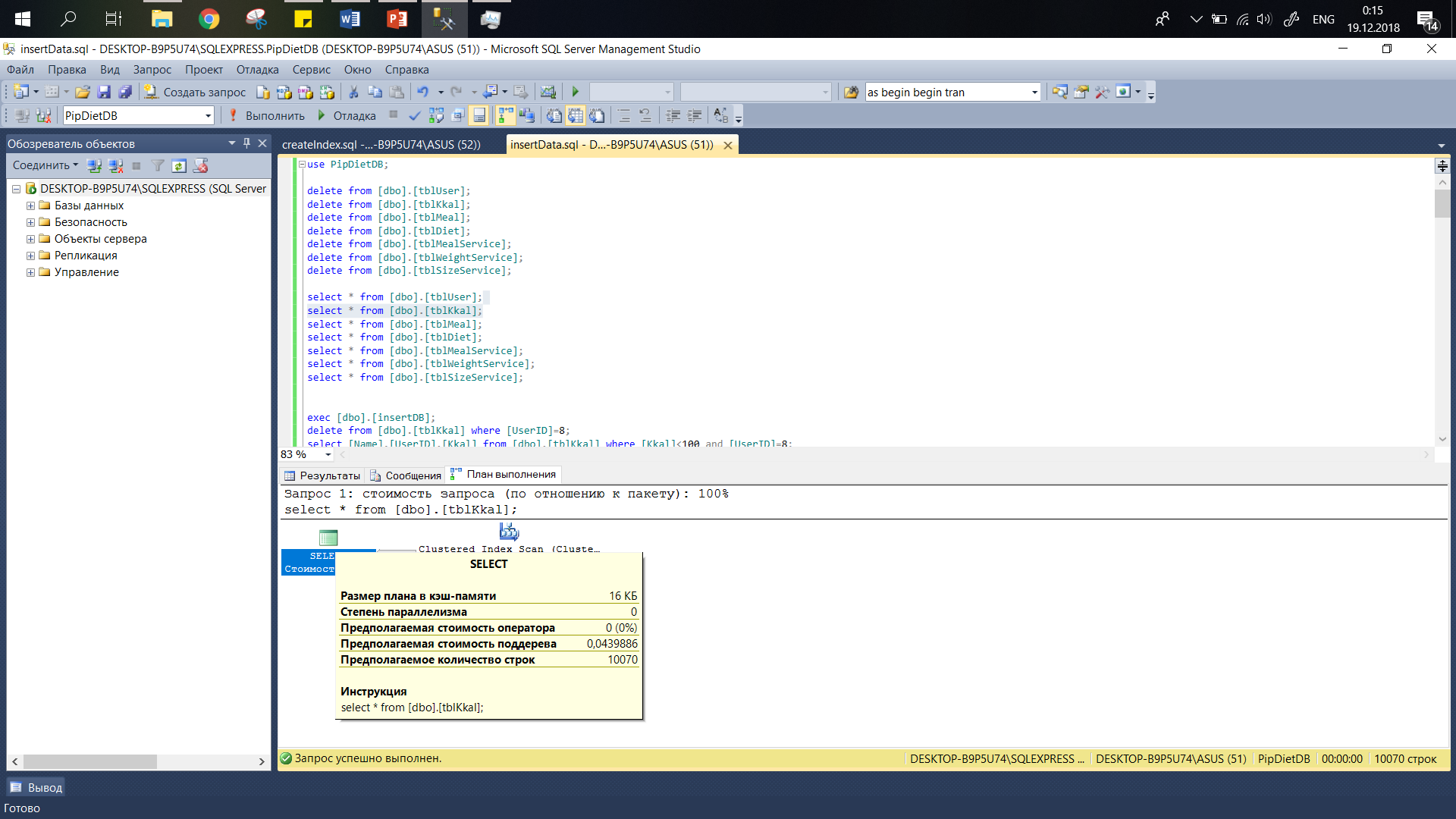


Рисунок 5.2 – Результат select-запроса с 10 000 строками

Далее проведем тестирования select-запроса на выборку автора и калорийности продуктов из 10 000 строк данных. Результаты представлены на рисунке 5.3.

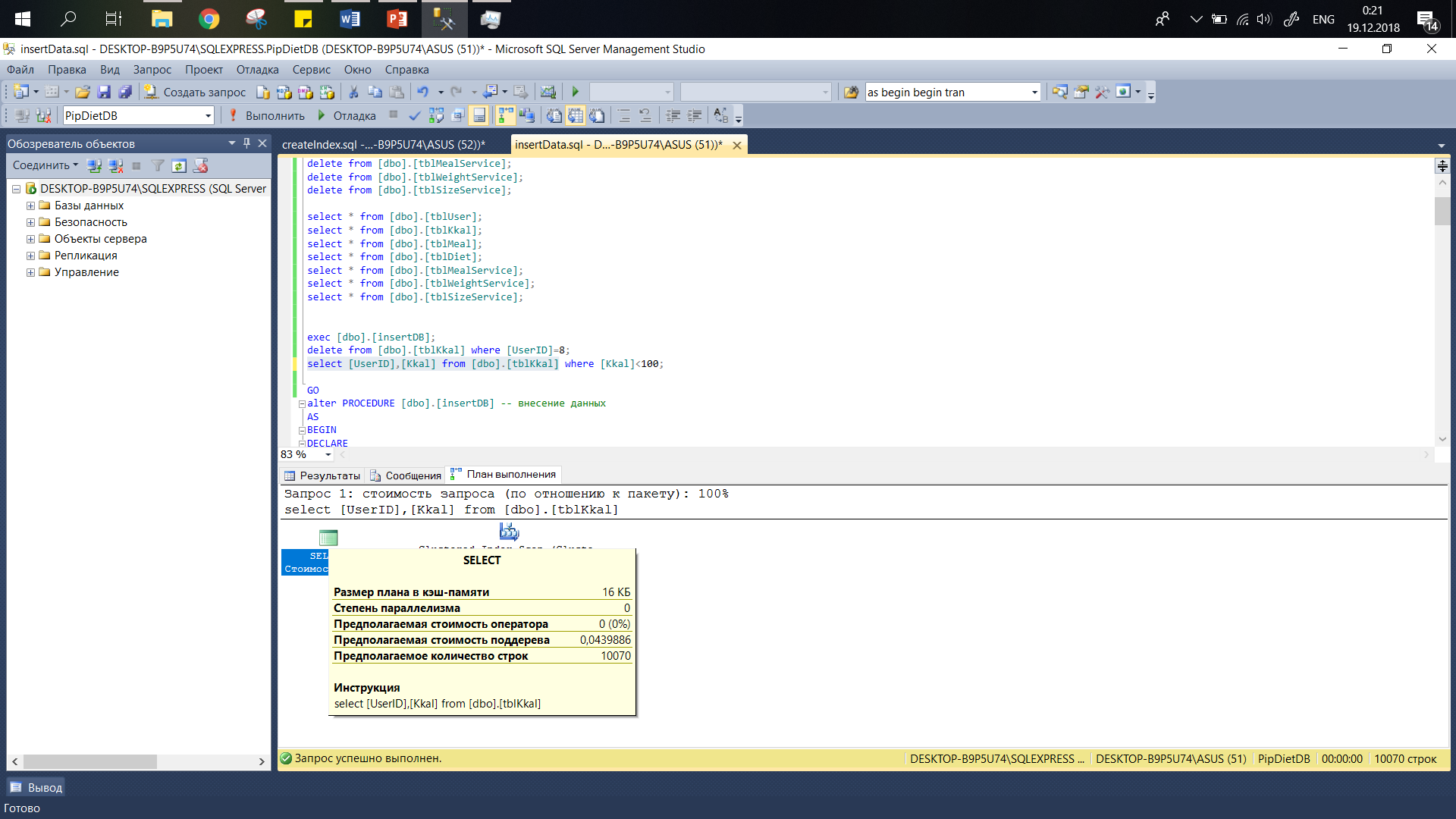


Рисунок 5.3 – Результат select-запроса

Здесь мы можем увидеть, что разницы в затраченном времени выборки по двум полям из таблицы продуктов или по всем – нет. Для оптимизации данного запроса был разработан некластеризованный индекс по таблице tblKkal, поля - UserID и Kkal. Благодаря данному индексу стоимость данного запроса уменьшила в более десяти раз. Данное утверждение подтверждается рисунком 5.4.

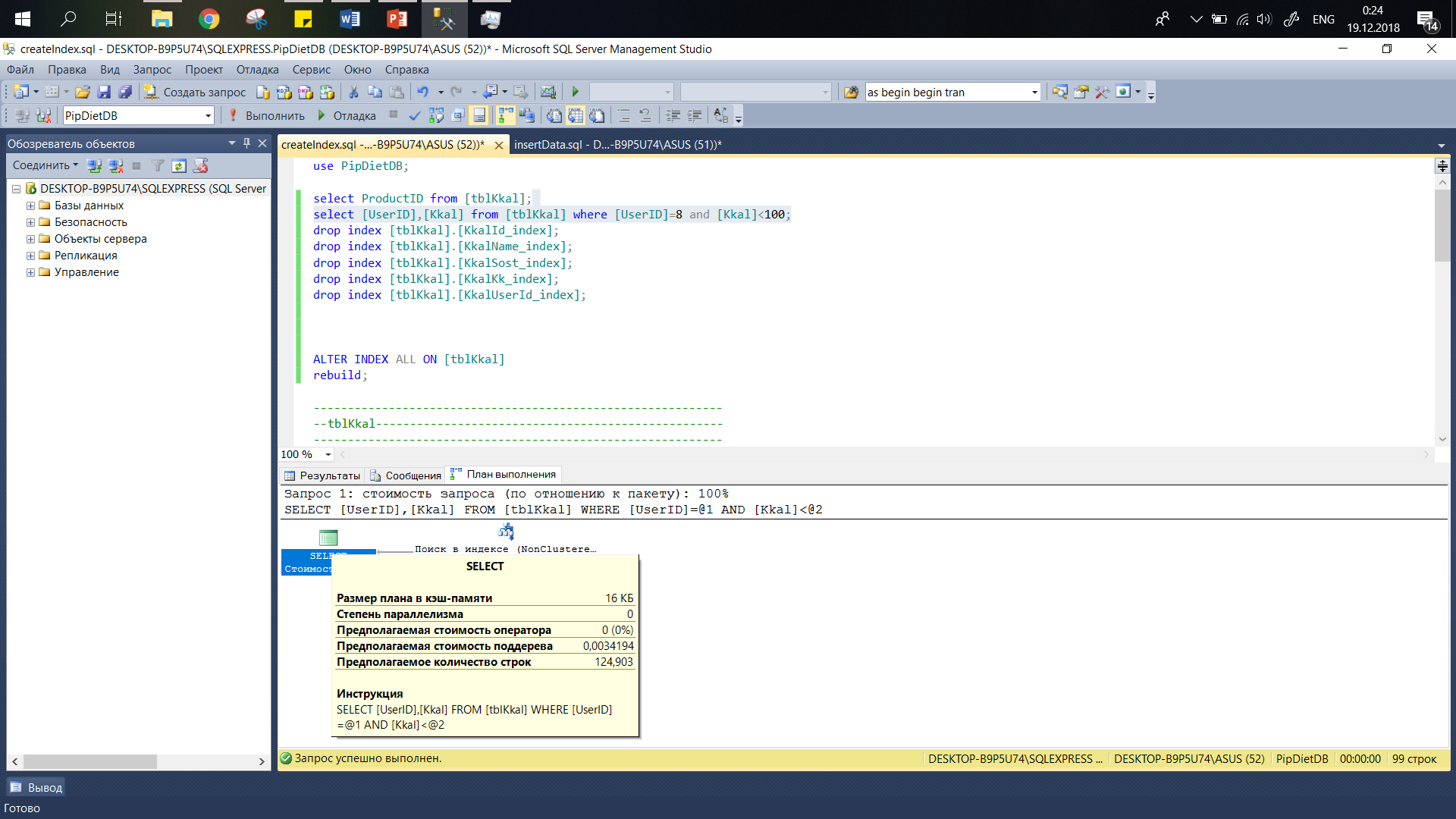


Рисунок 5.4 – Результат select-запроса с индексом

В ходе тестирования приложения были рассмотрены ситуации, при которых пользователь мог ввести данные неверно. Примером подобного действия будет неверный ввод имени и пароля для пользователя или админа. Предупреждение, возникающее на это действие, будет выглядеть следующим образом (рисунок 5.6).

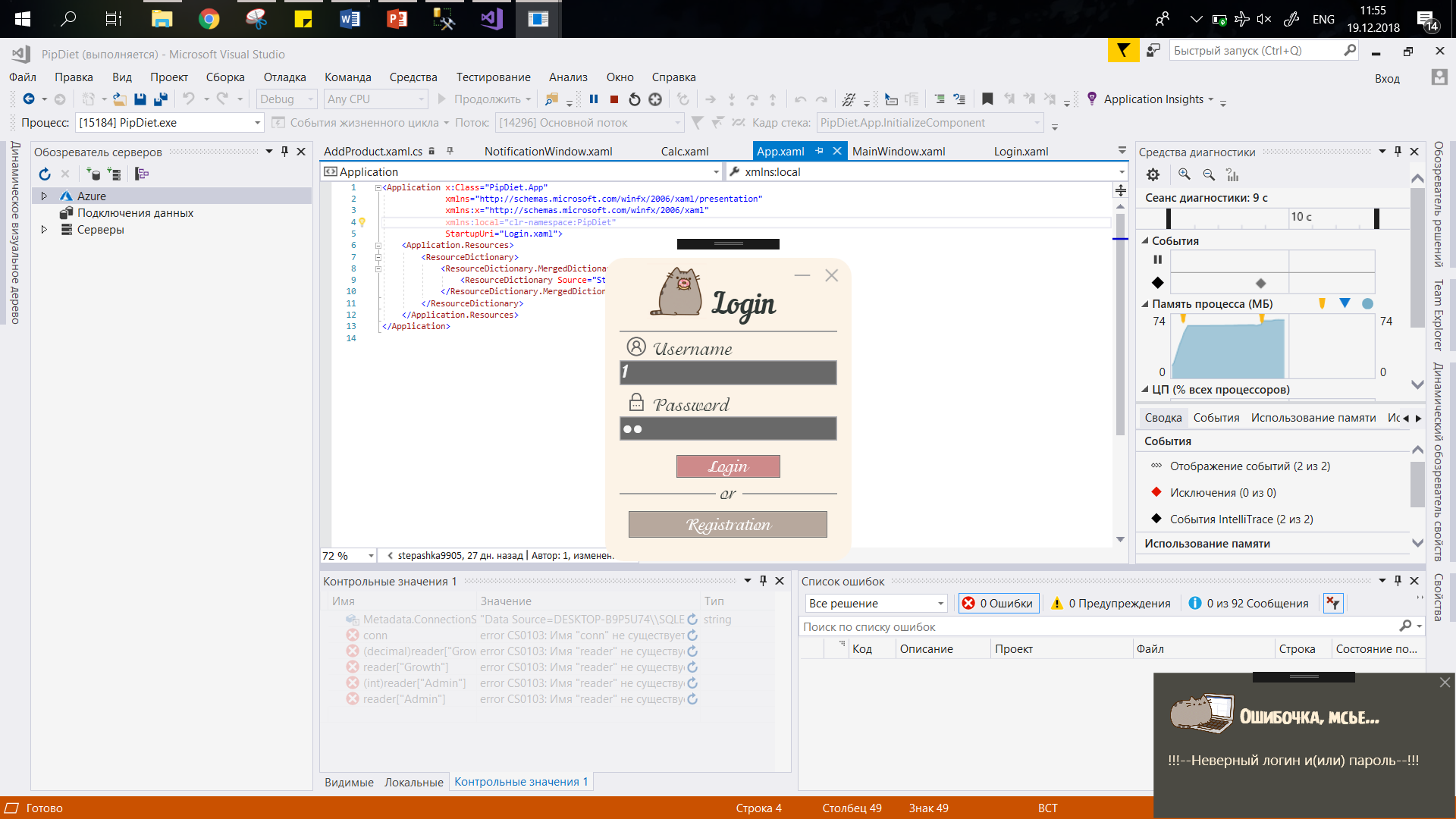


Рисунок 5.5 – Предупреждение при неверном входе

Аналогичным образом пользователь может получать предупреждения о неверном вводе данных и в других частях приложения.

После добавления 10 000 данных, было проведено тестирование приложения и сделаны следующие выводы: при таком большом объеме денных приложение нуждается в доработке, т.к. при выборке такого большого количества продуктов, оно становится не дееспособным. Один из вариантов решения данной проблемы – постраничная загрузка данных, примерно по 100, т.к. при таком объеме проблем не наблюдалось.

# Заключение

В процессе выполнения курсовой работы была спроектирована база данных, предметной областью которой является система управления режимом питания. Также было разработано приложение демонстрирующее работу с базой данных PipDietDB. База данных была разработана в рамках Microsoft SQL Server – система управления реляционными базами данных. Данная РСУБД используется для работы с базами данных размером от персональных до крупных баз данных масштаба предприятия.

Одной из главных задач данной курсовой работы - освоение технологии SQL Server Reporting Services и данная цель была достигнута. В рамках курсовой работы было разработано порядка 6 различных отчетов, построенных на базе данных PipDietDB. Данная технология довольно удобна для построения различных отчетов на базах данных.

# Список использованных источников

1. Блинова Е.А. Курс лекций по базам данных / Е.А. Блинова

2. Microsoft OLE DB [Электронный ресурс]/ msdn.microsoft.com – Режим доступа: https://msdn.microsoft.com/en-us/library/ms722784(v=vs.85).aspx Дата доступа 20.11.2018.

3. Пацей, Н.В. Технология разработки программного обеспечения / Н.В. Пацей. – Минск: БГТУ, 2016. – 129 с.

5. ProfessorWeb .NET & Web Programming [Электронный ресурс] / Режим доступа: <https://professorweb.ru> – Дата доступа: 31.10.2018.

# ПРИЛОЖЕНИЕ А

Процедура для добавления нового пользователя в базу данных.

GO

ALTER procedure [dbo].[AddNewUser]

(

@username nvarchar(50), --логин

@password nvarchar(50), --пароль

@admin int,--роль

@sex nvarchar(6), --пол

@birth int, --возраст

@growth decimal(18,0), --рост

@idealweight decimal(18,0), --идеальный вес

@calories1 int, --норма калорий 1

@calories2 int, --норма калорий 2

@calories3 int --норма калорий 3

)

as begin

begin try

begin tran

insert into [dbo].[tblUser]([Username],[Password],[Admin],[Sex],[Birthday],[Growth],[IdealWeight],[DailyCaloriesNormEasy],[DailyCaloriesNormNorm],[DailyCaloriesNormHard])

values (@username,@password,@admin,@sex,@birth,@growth,@idealweight,@calories1,@calories2,@calories3);

commit;

end try

begin catch

print error\_message()

end catch

end

# Приложение Б

Формула расчёта BMR – суточной нормы калорий для человека (формула Миффлина-Сан Жеора).

Коэффициент физической нагрузки (КФН):

1. минимальная: КФН = 1.2;
2. низкая: КФН = 1.375;
3. средняя: КФН = 1.55;
4. высокая: КФН = 1.725;
5. очень высокая: КФН = 1.9.

Коэффициент, зависящий от цели человека (КЦ):

1. похудание: КЦ = 0.8;
2. поддержка веса: КЦ = 1;
3. набор массы: КЦ = 1.2.

BMR для женщин:

BMR = (447.6 + (9.2 x вес, кг) + (3.1 х рост, cм) – (4.3 х возраст, лет)) \* КФН.

BMR для мужчин:

BMR = (88.36 + (13.4 x вес, кг) + (4.8 х рост, см) – (5.7 х возраст, лет)) \* КФН.

Формула расчёта идеального веса для человека (формула Купера).

Идеальный вес для женщины (кг): (рост(см) х 3,5/2,54-108) х 0,453;

Идеальный вес для мужчины(кг): (рост(см) х 3,5/2,54-128) х 0,453;