|  |
| --- |
| Пермский филиал федерального государственного автономного образовательного учреждения высшего образования  «Национальный исследовательский университет  «Высшая школа экономики»    [*Факультет социально-экономических и компьютерных наук*](https://perm.hse.ru/scs/) |
| Белов Егор Александрович  **РАЗРАБОТКА ПРИЛОЖЕНИЯ ДЛЯ РАСЧЕТА КАЛОРИЙНОСТИ ПРОДУКТОВ И БЛЮД**  *Курсовая работа*  студента образовательной программы «Программная инженерия» по направлению подготовки *09.03.04 Программная инженерия*  Руководитель  к.ф.-м.н., доцент, доцент кафедры информационных технологий в бизнесе  Е.Б. Замятина |

Пермь, 2023 год

**Аннотация**

Автор: Белов Егор Александрович.

Название работы: Разработка приложения для расчета калорийности продуктов и блюд.

Данная работа содержит процесс анализа и разработки приложения для расчета калорийности продуктов и блюд. Курсовая работа содержит 44 страницы и 5 глав. В данной работе содержится 5 рисунков, 5 приложений.

Оглавление

[Введение 4](#_Toc130586519)

[Глава 1. Анализ предметной области 6](#_Toc130586520)

[1.1 Назначение системы и основные характеристики 6](#_Toc130586521)

[1.2 Описание аналогов 7](#_Toc130586522)

[1.2.1 MyFitnessPal 7](#_Toc130586523)

[1.2.2 Yazio 8](#_Toc130586524)

[1.2.3 FatSecret 10](#_Toc130586525)

[1.2.4 Онлайн сервисы 10](#_Toc130586526)

[1.2.5 Итоги 11](#_Toc130586527)

[1.3 Описание бизнес-процессов 11](#_Toc130586528)

[1.4 Описание прецедентов 12](#_Toc130586529)

[1.5 Требования к системе 16](#_Toc130586530)

[Глава 2. Технологии и выбор средств реализации приложения 17](#_Toc130586531)

[2.1 Выбор СУБД для базы данных 17](#_Toc130586532)

[2.2 Выбор программных средств реализации 18](#_Toc130586533)

[Глава 3. Проектирование Базы данных 19](#_Toc130586534)

[3.1 Нормализация 19](#_Toc130586535)

[3.1.1 Функциональные зависимости 19](#_Toc130586536)

[3.1.2 Приведение в 1НФ 19](#_Toc130586537)

[3.1.3 Приведение в 2 НФ 22](#_Toc130586538)

[3.1.4 Приведение в 3 НФ 23](#_Toc130586539)

[3.2 Описание таблиц и их составляющих 24](#_Toc130586540)

[Глава 4. Проектирование интерфейса 27](#_Toc130586541)

[Глава 5. Реализация приложения 30](#_Toc130586542)

[5.1 Описание сущностей 30](#_Toc130586543)

[5.2 Реализация регистрации 32](#_Toc130586544)

[5.3 Реализация авторизации 33](#_Toc130586545)

[5.4 Реализация страниц 34](#_Toc130586546)

[5.4.1 Реализация сервиса Приемов пищи 34](#_Toc130586547)

[5.4.2 Реализация основной страницы 35](#_Toc130586548)

[5.5 Верстка страниц 38](#_Toc130586549)

[Заключение 39](#_Toc130586550)

[Библиографический список 40](#_Toc130586551)

[Приложение А. Диаграмма прецедентов 41](#_Toc130586552)

[Приложение Б. ER-диаграмма базы данных 42](#_Toc130586553)

[Приложение C. SQL-запросы 43](#_Toc130586554)

[Приложение D. Код программы 44](#_Toc130586555)

Введение

Данная работа посвящена созданию приложения для расчета калорийности продуктов и блюд.

В современном мире всё большее количество людей проявляют интерес к своему здоровью и питанию, понимая, что правильное питание играет важную роль в сохранении здоровья и повышении качества жизни. Люди стали заботиться о своем питании по ряду причин:

* Рост информированности. С развитием интернета и социальных сетей доступ к информации об оздоровительном питании стал значительно проще. Люди могут быстро и легко получить информацию о пищевых продуктах, их пользе и вреде, а также о правильном и здоровом питании в целом.
* Распространение тенденции к здоровому образу жизни. В настоящее время все больше людей стремятся соблюдать здоровый образ жизни, и питание является одним из главных аспектов здорового образа жизни. Люди понимают, что правильное питание помогает укрепить иммунитет, улучшить работу организма и предотвратить развитие многих заболеваний.
* Рост числа людей с аллергиями и непереносимостью к определенным продуктам. В современном мире все больше людей страдают от аллергий и непереносимости к определенным продуктам, что приводит к необходимости тщательно следить за своим питанием и выбирать продукты, которые не вызывают негативных реакций.
* Рост числа вегетарианцев и веганов. В настоящее время все больше людей отказываются от мяса и других животных продуктов, становясь вегетарианцами или веганами. Это требует тщательного подхода к питанию и поиска альтернативных источников необходимых питательных веществ.
* Рост числа людей, занимающихся спортом: Спортсмены и люди, которые занимаются фитнесом или другими видами активности, нуждаются в правильном питании для поддержания своего здоровья и физической формы.

Все эти факторы влияют на то, что в современном мире все больше людей стали заботиться о своем питании и выбирают продукты более осознанно, учитывая их пользу для здоровья. Один из ключевых аспектов здорового питания – контроль за калорийностью потребляемых продуктов и блюд.

Объектом исследования является данная предметная область, то есть информация о продуктах, блюдах и их калорийности.

Предметом исследования является приложение, которое позволяет вести дневник приемов пищи и получать информацию о потребляемых калориях.

Целью данной работы является разработка приложения для расчета калорийности продуктов и блюд.

Для достижения поставленной цели требуется выполнить ряд задач:

* Провести анализ предметной области, выявить бизнес-процессы и сформулировать функциональные и нефункциональные требования к системе.
* Выбрать инструменты для разработки информационной системы.
* Спроектировать и нормализовать базу данных на основе анализа предметной области.
* Разработать дизайн приложения.
* Реализовать приложение.
* Провести тестирование приложения различными методами.
* Отладить программу, исправив ошибки, выявленные в процессе тестирования.

В ходе разработки программной системы был выполнен анализ предметной области, проанализированы существующие приложения с определением их достоинств и недостатков. На основании полученных результатов были спроектированы приложение и интерфейс, а затем разработаны с учетом всех функциональных и нефункциональных требований.

Глава 1. Анализ предметной области

На этапе анализа предметной области проводится изучение объектов и особенностей данной области, а также определение связей между ними. Дополнительно производится анализ существующих аналогов программы с целью выявления их преимуществ и недостатков, которые учитываются при проектировании и разработке новой программы.

* 1. Назначение системы и основные характеристики

Предметной областью разрабатываемой базы данных является разработка приложения для расчета калорийности продуктов и блюд. Основной смысл приложения в структуризации приемов пищи и отслеживанием необходимых и потребляемых калорий в различных продуктах.

Каждый пользователь приложения – зарегистрированный пользователь, который может добавлять себе приемы пищи и следить за потребляемыми калориями.

У пользователя есть возможность вести дневник веса, чтобы следить за динамикой изменений массы тела.

Пользователь имеет возможность создавать новые блюда и ингредиенты.

Пользователь имеет доступ к просмотру информации о продуктах.

Назначение системы:

Цель приложения для подсчета калорий в продуктах и блюдах - помочь людям отслеживать ежедневное потребление калорий и поддерживать здоровое и сбалансированное питание. Приложение позволяет пользователям вводить информацию о продуктах и блюдах, которые они потребляют, а затем рассчитывает общее количество потребленных калорий. Отслеживая потребление калорий, пользователи могут лучше понять, как их диетический выбор влияет на их общее здоровье и самочувствие.

Основные функции:

* Приложение должно иметь полную базу данных продуктов и блюд, включая информацию об их питательности и количестве калорий.
* Приложение должно позволять пользователям отслеживать потребление калорий в течение дня и предоставлять сводку общего количества потребленных калорий.
* Приложение должно позволять пользователям создавать профиль с их личной информацией, включая возраст, пол, рост и вес, чтобы помочь рассчитать рекомендуемое ежедневное потребление калорий.
* Приложение должно позволять пользователям устанавливать цели по потреблению калорий, исходя из их задач по снижению или увеличению веса.
* Калькулятор рецептов: Приложение должно позволять пользователям вводить рецепты и рассчитывать общее потребление калорий на порцию.
  1. Описание аналогов

В данном разделе будут рассмотрены имеющиеся аналоги Приложения для расчета калорийности продуктов и блюд.

* + 1. MyFitnessPal

MyFitnessPal — это мобильное приложение, которое помогает пользователям отслеживать их диету, физические упражнения и прогресс в снижении веса. У этого приложения есть свои преимущества и недостатки.

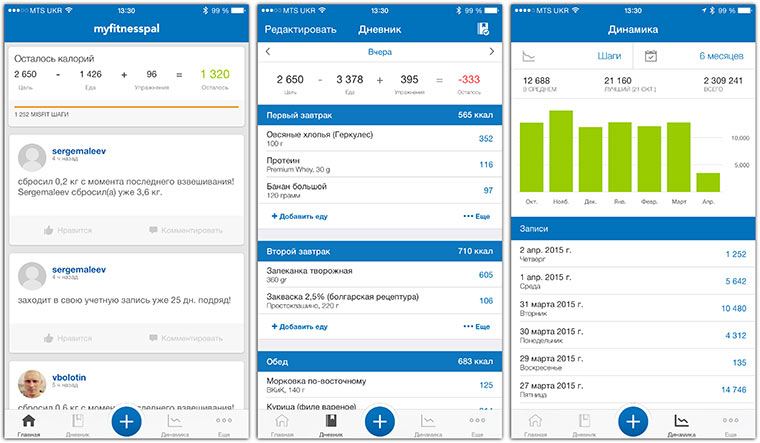
Преимущества:

* MyFitnessPal имеет одну из самых больших баз данных продуктов питания среди всех приложений для снижения веса.
* Приложение оснащено сканером штрих-кода, который может сканировать штрих-код упакованных продуктов, что облегчает отслеживание калорий и информации о питании.
* Приложение позволяет пользователям устанавливать собственные цели по снижению веса и настраивать потребление калорий в зависимости от уровня активности, веса и роста.
* MyFitnessPal имеет функцию сообщества, которая позволяет пользователям общаться с друзьями и делиться своим опытом похудения.

Недостатки:

* Приложение показывает рекламу и предлагает премиум-функции, требующие подписки.
* Хотя база данных приложения большая, некоторая информация в ней может быть неточной.
* Приложение не предоставляет углубленный анализ питания, что может не удовлетворить потребности пользователей.

На рисунке 1.1 представлен интерфейс приложения MyFitnessPal.



***Рисунок 1.1. Приложение MyFitnessPal***

В целом, MyFitnessPal может быть полезным инструментом для пользователей, которые хотят отслеживать потребление калорий и прогресс в потере веса, но имеются и недостатки.

* + 1. Yazio

Yazio — это мобильное приложение, которое помогает пользователям отслеживать потребление калорий, физические упражнения и прогресс в похудении. У приложения есть свои преимущества и недостатки.

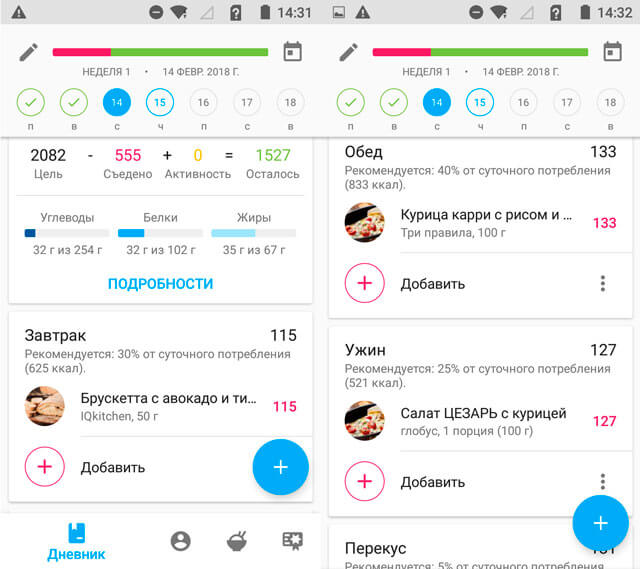
Преимущества:

* Yazio предлагает персонализированные планы потери веса, основанные на целях пользователя, уровне активности и диетических предпочтениях.
* Приложение имеет большую базу данных продуктов питания, включая ресторанные и фирменные продукты.
* Приложение оснащено сканером штрих-кода, который может сканировать штрих-код упакованных продуктов.
* Yazio не показывает никакой рекламы, обеспечивая беспроблемный пользовательский опыт.

Недостатки:

* Возможности приложения ограничены по сравнению с другими приложениями для похудения.
* Некоторые расширенные функции приложения, такие как планирование питания и отслеживание питательных веществ, требуют премиум-подписки.
* Хотя база данных приложения является полной, некоторая информация в приложении может быть неточной.

На рисунке 1.2 представлен интерфейс приложения Yazio.



***Рисунок 1.2. Приложение Yazio***

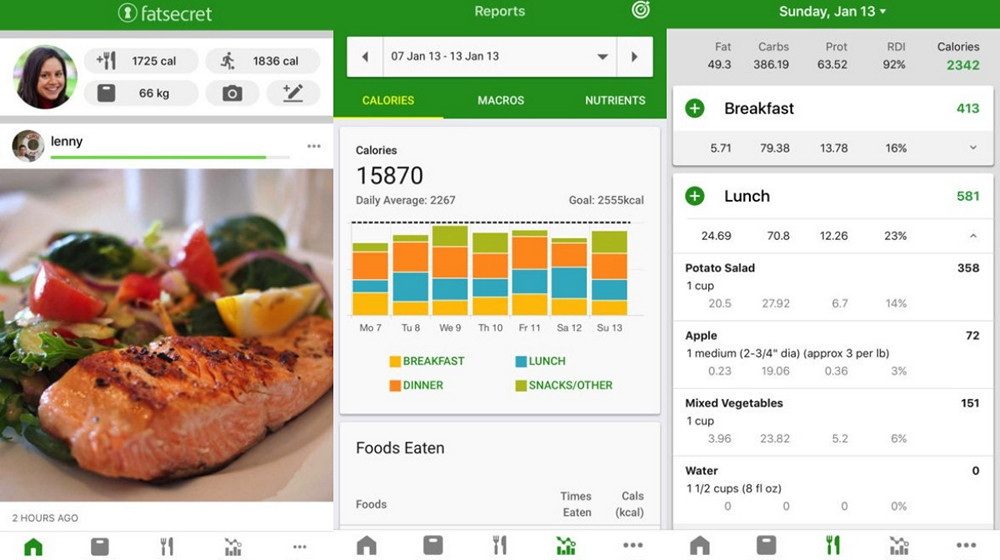
В целом, Yazio может быть полезным инструментом для пользователей, которые хотят отслеживать потребление калорий и прогресс в потере веса, но оно может не подойти всем из-за своих ограничений и неточной информации.

* + 1. FatSecret

FatSecret — это мобильное приложение, которое помогает пользователям отслеживать ежедневное потребление калорий, физические упражнения и прогресс в снижении веса. У приложения имеет как преимущества, так и недостатки.

Преимущества и недостатки данного приложения аналогичны с предыдущими примерами. Также приложения является бесплатным, несмотря на присутствие рекламы.

На рисунке 1.3 представлен интерфейс приложения FatSecret.



***Рисунок 1.3. Приложение FatSecret***

* + 1. Онлайн сервисы

Онлайн-калькуляторы калорий — это инструменты, которые помогают пользователям рассчитать количество калорий, которое им необходимо потреблять для достижения своих целей по снижению или увеличению веса.

Преимущества данных сервисов:

* Онлайн-калькуляторы калорий удобны в использовании и доступны из любого места, если есть подключение к Интернету.
* Большинство онлайн-калькуляторов калорий позволяют пользователям настраивать свои результаты на основе возраста, пола, веса, роста и уровня активности, что обеспечивает более точный подсчет калорий.
* Онлайн-калькуляторы калорий могут быть образовательными, поскольку они предоставляют пользователям информацию о том, сколько калорий им необходимо потреблять для поддержания, снижения или набора веса, а также о питательной ценности продуктов, которые они едят.
* Большинство онлайн-калькуляторов калорий являются бесплатными, что делает их доступными для широкой аудитории.

Недостатки:

* Результаты, предоставляемые онлайн-калькуляторами калорий, могут быть неточными для всех пользователей, так как они основаны на общих формулах и не учитывают индивидуальные факторы.
* Онлайн-калькуляторы калорий могут предоставлять подсчет калорий без учета питательной ценности продуктов или диетических ограничений пользователя.
* Некоторые пользователи могут посчитать объем информации, предоставляемой онлайн-калькуляторами калорий, чрезмерным.

В целом, онлайн-калькуляторы калорий могут быть полезным инструментом для некоторых пользователей, чтобы отслеживать потребление калорий и достигать своих целей по снижению или увеличению веса, но они не могут быть точными или подходящими для всех.

* + 1. Итоги

Существует множество различных аналогов Приложения для расчета калорийности продуктов и блюд. Основными преимуществами являются: большие базы данных, встроенные сканеры штрих-кодов, возможность выставления личной цели в похудении. Основными недостатками являются: наличие платной подписки для расширения функционала, неточность в объемной базе данных, мультиплатформенность.

* 1. Описание бизнес-процессов

Для реализации приложения необходимо выявить некоторые бизнес-процессы, то есть алгоритмы, которые повторяются в ходе взаимодействия пользователя с системой.

Были выявлены следующие бизнес-процессы:

* Авторизация и регистрация пользователей.
* Просмотр, редактирование, удаление, создание объектов, то есть, должна быть возможность гибкого взаимодействия пользователя с базой данных, например, создание нового блюда, редактирование информации о пользователе.
* Экспорт отчета в специальные форматы, то есть сохранение в отдельном файле определенной информации.
  1. Описание прецедентов

Далее будет разработана диаграмма прецедентов, то есть модель, которая описывает функционал приложения и результат взаимодействия пользователя с данной системой. Описание прецедентов представлено в таблицах 1.1 – 1.6.

***Таблица 1.1. Описание прецедента авторизации***

|  |  |
| --- | --- |
| Акторы | Пользователь, Администратор |
| Краткое описание | Авторизация пользователя при заходе в приложение |
| Триггер | Открытие приложения |
| Основной поток | 1. Система просит пользователя ввести почту и пароль. 2. Система проверяет данные на корректность. 3. Система проверяет наличие данных в базе данных. 4. Все прошло успешно, пользователь получает доступ к приложению. |
| Альтернативный поток | 1. Если введенные данные некорректны, то страница обновится, и пользователь получит возможность ввести данные еще раз. 2. Аналогично, если введенных данных нет в базе данных. |
| Результат | Пользователь получает доступ к приложению. |

***Таблица 1.2. Описание прецедента регистрации***

|  |  |
| --- | --- |
| Акторы | Пользователь, Администратор |
| Краткое описание | Регистрация нового пользователя при заходе в систему |
| Триггер | Открытие приложения и нажатие кнопки Регистрация |
| Основной поток | 1. После нажатия на кнопку регистрации, приложение перенаправит пользователя на специальную страницу регистрации. 2. Система просит пользователя ввести почту, пароль и другие характеристики, такие как пол, возраст, рост, вес. 3. Система проверит введенные данные 4. Если данные корректны, система создаст новый объект в базе данных. 5. Система перенаправит пользователя на главную страницу приложения. |
| Альтернативный поток | 1. Если введенные данные некорректны или данная почта уже находится в базе данных, то страница обновится, и пользователь получит возможность ввести данные еще раз. |
| Результат | Пользователь получает доступ к приложению. |

***Таблица 1.3. Описание прецедента удаления***

|  |  |
| --- | --- |
| Акторы | Пользователь, Администратор |
| Краткое описание | Удаления приема пищи |
| Триггер | Нажатие кнопки удалить на элементе конкретного приема пищи |
| Основной поток | 1. Система, после нажатия на кнопку удалить у конкретного объекта, определяет идентификатор этого объекта. 2. Система проверяет наличие объекта с данным идентификатором в базе данных. 3. Если объект найден, объект удалится из базы данных. 4. Пользователь увидит обновленную страницу без данного объекта. |
| Альтернативный поток | 1. Если объект не найден, пользователь увидит сообщение об ошибке. 2. Приложение перезагрузится. |
| Результат | Пользователь увидит обновленную страницу без данного объекта. |

Примечание: В таблице 1.3 описан прецедент удаления приема пищи. Для остальных сущностей и объектов процесс удаления аналогичен.

***Таблица 1.4. Описание прецедента создания***

|  |  |
| --- | --- |
| Акторы | Пользователь, Администратор |
| Краткое описание | Создание приема пищи |
| Триггер | Нажатие кнопки добавить на странице всех приемов пищи |
| Основной поток | 1. Система, после нажатия на кнопку добавить, загрузить новую специальную страницу создания приема пищи. 2. Пользователю будет дана возможность ввести характеристики объекта. 3. Система проверит введенные данные. 4. Система создаст новый объект в базе данных. 5. Пользователь увидит созданный объект на страницу Приемы пищи. |
| Альтернативный поток | 1. Если введенные данные некорректны, то страница обновится, и пользователь получит возможность ввести данные еще раз. |
| Результат | Пользователь увидит созданный объект на странице Приемы пищи. |

Примечание: В таблице 1.4 описан прецедент создания приема пищи. Для остальных сущностей и объектов процесс создания аналогичен.

***Таблица 1.5. Описание прецедента редактирования***

|  |  |
| --- | --- |
| Акторы | Пользователь, Администратор |
| Краткое описание | Изменение приема пищи |
| Тригер | Нажатие кнопки редактировать на элементе конкретного объекта |
| Основной поток | 1. Система, после нажатия на кнопку редактировать, сохранить идентификатор выбранного объекта. 2. Система перенаправит пользователя на страницу редактирования. 3. Пользователю будет дана возможность ввести характеристики объекта. 4. Система проверит введенные данные. 5. Система создаст новый объект по введенным данным. 6. По идентификатору найдет выбранный объект в базе данных. 7. Заменит поля выбранного объекта на поля нового объекта. 8. Система сохранит обновленный объект в базу данных и перенаправит пользователя на главную страницу приложения. |
| Альтернативный поток | 1. Если введенные данные некорректны, то страница обновится, и пользователь получит возможность ввести данные еще раз. 2. Если объект не найден, пользователь увидит сообщение об ошибке. 3. Приложение перезагрузится. |
| Результат | Пользователь увидит измененный объект на странице Приемы пищи. |

Примечание: В таблице 1.5 описан прецедент редактирования приема пищи. Для остальных сущностей и объектов процесс редактирования аналогичен.

***Таблица 1.5. Описание прецедента экспорт***

|  |  |
| --- | --- |
| Акторы | Пользователь, Администратор |
| Краткое описание | Экспорт определенных данных |
| Триггер | Нажатие кнопки экспорт на определенной странице |
| Основной поток | 1. После нажатия на кнопку, система создаст файл с отображаемой на текущей странице таблицей |
| Результат | У пользователя появился новый файл |

* 1. Требования к системе

Список эксплуатационных спецификаций, состоящий из функциональных и нефункциональных требований, описывающий системные требования к разрабатываемому приложению.

Функциональные требования:

* Приложение должно позволять пользователям зарегистрироваться и создать учетную запись, указав логин, пароль, имя, вес, рост, возраст, пол и уровень физической активности.
* Приложение должно позволять пользователям записывать свой текущий вес в дневник веса с указанием даты измерения.
* Приложение должно иметь возможность рассчитывать калорийность отдельных продуктов на основе их пищевой ценности, а также калорийность блюд на основе их ингредиентного состава.
* Приложение должно позволять пользователям искать продукты в базе данных по названию или категории, а также просматривать их калорийность и информацию о питании.
* Приложение должно позволять пользователям создавать блюда, выбирая ингредиенты из базы данных и указывая их количество.
* Приложение должно позволять пользователям отслеживать потребленные ими блюда с указанием даты и времени потребления.
* Приложение должно уметь рассчитывать общую калорийность рациона пользователя на основе его веса, роста, возраста, пола, уровня физической активности, а также потребленных им блюд.

Нефункциональные требования:

* Приложение должно обеспечивать безопасность пользовательских данных, включая информацию о логине и пароле, путем применения механизмов шифрования и безопасного хранения.
* Приложение должно быть способно работать с большим количеством пользователей и растущей базой данных продуктов и блюд.
* Приложение должно быть удобным и интуитивно понятным, с четкой и простой навигацией и визуально привлекательным интерфейсом.

Глава 2. Технологии и выбор средств реализации приложения

В этой главе мы рассмотрим и обсудим инструменты и технологии, которые могут быть использованы для создания клиентских приложений и удаленной базы данных.

2.1 Выбор СУБД для базы данных

PostgreSQL — это мощная реляционная система управления базами данных (РСУБД) с открытым исходным кодом, которая широко используется для хранения и управления данными. Она известна своей надежностью, стабильностью и универсальностью, что делает ее популярным выбором для многих разработчиков и организаций.

Некоторые преимущества использования PostgreSQL включают:

* PostgreSQL может обрабатывать большие и сложные наборы данных, что делает его подходящим для использования в приложениях, требующих высокой масштабируемости.
* PostgreSQL обладает высокой надежностью, а такие функции, как транзакционная целостность и механизмы восстановления после сбоев, обеспечивают целостность данных и минимизируют риск их потери.
* PostgreSQL поддерживает широкий спектр типов данных и имеет гибкую архитектуру, которая позволяет пользователям настраивать и расширять ее в соответствии со своими потребностями.
* PostgreSQL оптимизирован для высокой производительности и может обрабатывать большое количество одновременных пользователей и транзакций.
* Будучи базой данных с открытым исходным кодом, PostgreSQL имеет большое и активное сообщество, которое вносит свой вклад в ее развитие, поддержку и обслуживание.
* PostgreSQL имеет встроенные функции безопасности, такие как контроль доступа, шифрование и механизмы аутентификации, которые помогают защитить данные от несанкционированного доступа и обеспечить соответствие нормам конфиденциальности данных.

В целом, PostgreSQL — это надежная, прочная и гибкая система управления базами данных, которая предлагает широкий спектр возможностей и преимуществ для разработчиков и организаций.

2.2 Выбор программных средств реализации

Было решено, что для серверной части проекта будет использоваться ASP.NET вместе с Entity Framework Core. Эта комбинация является проверенным и надежным решением, которое обещает отличную производительность и надежную функциональность.

ASP.NET и Entity Framework Core — это основа веб-разработки, которая объединяет преимущества платформы .NET, основы веб-приложений ASP.NET и библиотеки Entity Framework Core ORM (Object-Relational Mapping). Она используется для создания современных и надежных веб-приложений, API и микросервисов, которые могут быть развернуты на широком спектре платформ и устройств.

Кроме того, создание пользовательского интерфейса будет осуществляться с помощью Razor Pages. Этот подход обеспечивает бесшовную интеграцию HTML, CSS и JavaScript с кодом на стороне сервера. Он обещает обеспечить элегантный и интуитивно понятный пользовательский интерфейс, который привлечет и заинтересует пользователей.

Данные технологии будут работать в гармонии для создания динамичного и эффективного веб-приложения.

Глава 3. Проектирование Базы данных

На этапе проектирования базы данных происходит учет необходимых сущностей и атрибутов, связи между ними, нормализация, приведение базы данных в 3 НФ.

3.1 Нормализация

3.1.1 Функциональные зависимости

При разработке баз данных и определении функциональных зависимостей между атрибутами необходимо учитывать следующие связи между ними:

1 ФЗ: По коду пользователя можно однозначно определить его Имя, Пол, Возраст, Вес, Рост, Уровень физической активности.

2 ФЗ: По коду пользователя можно однозначно определить его Почту, Пароль для входа в аккаунт.

3 ФЗ: По коду дневника веса можно определить его Пользователя, Дату, Вес.

4 ФЗ: По коду физической активности можно однозначно определить Описание, Оценку активности.

5 ФЗ: По коду приемов пищи можно однозначно определить Тип приема пищи, Дату, Принимаемое блюдо, Пользователя.

6 ФЗ: По коду типа приема пищи можно определить Название типа, Описание.

7 ФЗ: По коду блюда можно определить Название и Категорию.

8 ФЗ: По коду категории блюда можно определить Название категории и Описание.

9 ФЗ: По коду продукта можно однозначно определить Название продукта, Категорию и Калорийность.

10 ФЗ: По коду категории продукта можно определить Наименование категории и Описание.

11 ФЗ: По коду ингредиента можно однозначно определить Продукт и Его вес, Блюдо.

3.1.2 Приведение в 1НФ

Отношение находится в первой нормальной форме (1НФ), если все его атрибуты (столбцы) содержат только атомарные (неделимые) значения и не допускают множественных значений или группировок. Другими словами, каждый столбец должен содержать только одно значение для каждой строки, а не набор значений или список значений.

В данном случае исходное **универсальное отношение** включает следующие атрибуты:

1. Код пользователя.
2. Имя пользователя.
3. Рост пользователя.
4. Вес пользователя.
5. Возраст пользователя.
6. Пол пользователя.
7. Код физической активности.
8. Код физической активности.
9. Описание физической активности.
10. Оценка физической активности.
11. Код пользователя.
12. Почта.
13. Пароль.
14. Код дневника веса.
15. Код пользователя.
16. Дата.
17. Вес.
18. Код приема пищи.
19. Код типа приема пищи.
20. Дата и время приема пищи.
21. Код блюда.
22. Код пользователя.
23. Код типа приема пищи.
24. Наименование.
25. Описание.
26. Код блюда.
27. Название блюда.
28. Код ингредиента.
29. Код категории блюда.
30. Код категории блюда.
31. Описание.
32. Наименование.
33. Код продукта.
34. Название продукта.
35. Код категории продукта.
36. Калорийность продукта.
37. Код категории продукта.
38. Наименование.
39. Описание.
40. Код ингредиента.
41. Код продукта.
42. Вес продукта.
43. Код блюда.

Согласно указанным выше функциональным зависимостям, определяем первичный ключ отношения, который состоит из следующих атрибутов:

* Код пользователя
* Код продукта
* Код блюда
* Код категории продукта
* Код категории блюда
* Код дневника веса
* Код типа приема пищи
* Код приема пищи
* Код физической активности
* Код ингредиента

Комбинация этих атрибутов обеспечивает уникальность каждой строки и позволяет однозначно идентифицировать запись. Чтобы гарантировать атомарность, необходимо в каждой строке таблицы указывать значения всех этих атрибутов, не группируя данные. С учетом вышесказанного, отношение соответствует первой нормальной форме.

3.1.3 Приведение в 2 НФ

2НФ достигается, когда каждый не ключевой атрибут в таблице зависит только от всего первичного ключа (ПК), а не от его составных частей. Если какой-либо не ключевой атрибут зависит только от части ПК, то необходимо разбить таблицу на несколько, чтобы каждая из них имела свой собственный ПК и соответствующие зависимости. Это позволяет устранить избыточность данных и снизить риск ошибок при изменении записей.

Имеется одиннадцать отношений (таблиц):

“Пользователи”, которое включает атрибуты: Код пользователя, Имя, Рост, Вес, Возраст, Пол, Код физической активности.

“Входные данные пользователей”, которое включает атрибуты: Код пользователя, Почта, Пароль.

“Физическая активность”, которое включает атрибуты: Код физической активности, Описание, Оценка физической активности.

“Дневник веса”, которое включает атрибуты: Код дневника веса, Код пользователя, Дата, Вес.

“Приемы пищи”, которое включает атрибуты: Код приема пищи, Код типа приема пищи, Дата, Код пользователя.

“Тип приема пищи”, которое включает атрибуты: Код типа приема пищи, Наименование, Описание.

“Блюда”, которое включает атрибуты: Код блюда, Название, Код категории блюда.

“Категории блюда”, которое включает атрибуты: Код категории блюда, Наименование, Описание.

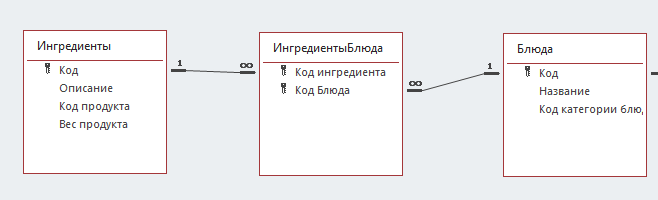
“Продукты”, которое включает атрибуты: Код продукта, Название, Код категории продукта, Калорийность.

“Категории продукта”, которое включает атрибуты: Код категории продукта, Наименование, Описание.

“Ингредиенты”, которое включает атрибуты: Код ингредиента, Описание, Код продукта, Вес продукта.

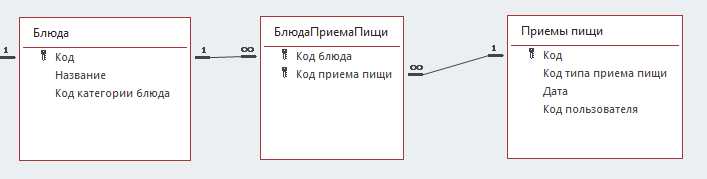
В приведённых отношениях отсутствуют частичные функциональные зависимости.

Необходимо реализовать связь “многие ко многим”. Данная связь имеется между отношениями “Приемы пищи” и “Блюда” (то есть во время приема пищи имеется возможность съесть несколько блюд, а соответственно одно блюдо, может быть, в нескольких приемах пищи), “Блюда” и “Ингредиенты” (блюдо состоит из нескольких ингредиентов, а также ингредиент, может быть, в нескольких блюдах). Для реализации необходимо создать дополнительные таблицы, состоящие из первичных ключей родительских таблиц. Схема таблицы ИнгредиентыБлюда представлена на рисунке 3.1.



***Рисунок 3.1. Схема, полученной таблицы ИнгредиентыБлюда***

Схема таблицы БлюдаПриемыПищи представлена на рисунке 3.2.



***Рисунок 3.2. Схема, полученной таблицы БлюдаПриемыПищи***

Таким образом, все отношения находятся в 2НФ.

3.1.4 Приведение в 3 НФ

Отношение находится в 3НФ, если оно находится во 2НФ и каждый не ключевой атрибут зависит только от Первичного Ключа (ПК) и не зависит от других не ключевых атрибутов. Если есть зависимость между не ключевыми атрибутами, то их следует вынести в отдельную таблицу. Это позволяет избежать транзитивных зависимостей между атрибутами и улучшить структуру БД.

В описанных выше отношениях “Пользователи”, “Входные данные пользователей”, “Физическая активность”, “Дневник веса”, “Приемы пищи”, “Тип приема пищи”, “Блюда”, “Категории блюда”, “Продукты”, “Категории продукта”, “Ингредиенты”, “БлюдаИнгрединеты”, “ПриемыПищиБлюда” отсутствуют зависимости между не ключевыми атрибутами, следовательно, они находятся в 3НФ. Схема итоговой базы данных представлена в Приложении D.

3.2 Описание таблиц и их составляющих

Далее в таблице 3.1 представлены существующие объекты, их атрибуты с характеристиками и примерами.

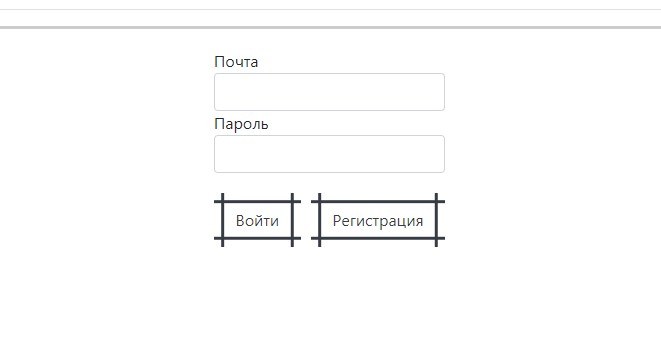
***Таблица 3.1 Физическая модель данных***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Объект | Атрибут | Вид атрибута | Свойства | Пример |
| Пользователь | Код пользователя Имя Рост Вес Код физической активности Пол Возраст | Счетчик Короткий текст Числовой Числовой Числовой Короткий текст Числовой | 6 символов 15 символов Вещественное Вещественное Целое 6 символов Вещественное | Автогенерация Егор 180 70 1 M 19 |
| Объект | Атрибут | Вид атрибута | Свойства | Пример |
| Входные данные пользователя | Код пользователя Почта Пароль | Счетчик Короткий текст Короткий текст | 6 символов 20 символов 20 символов | Автогенерация e1@mail.ru 123 |
| Дневник веса | Код дневника веса Дата Вес Код пользователя | Счетчик Дата и время Числовой Числовой | 6 Символов  Вещественное Целое | Автогенерация 19.03.2023 70 0 |
| Физическая активность | Код физической активности Описание Оценка | Счетчик  Короткий текст Числовой | 6 символов  20 символов Вещественное | Автогенерация  Активные тренировки 1.31 |
| Приемы пищи | Код приема пищи Код типа приема пищи Дата Код пользователя Код блюда | Счетчик Числовой Дата и время Числовой Числовой | 6 символов Целое  Целое Целое | Автогенерация 1 20.03.2023 15:41 1 1 |
| Типы приемов пищи | Код типа приема пищи Наименование Описание | Счетчик Короткий текст Короткий текст | 6 символов 20 символов 20 символов | Автогенерация Завтрак Первый прием пищи |
| Блюда | Код блюда Название Код приема пищи Код категрии блюда | Числовой Короткий текст Числовой Числовой | 6 символов 15 символов 6 символов Целое | Автогенерация Яичница 1 1 |
| Категории блюда | Код категории блюда Наименование Описание | Счетчик Короткий текст Короткий текст | 6 символов 20 символов 20 символов | Автогенерация Жареное Пожарили |
| Объект | Атрибут | Вид атрибута | Свойства | Пример |
| БлюдаПриемыПищи | Код приема пищи Код блюда | Числовой Числовой | Целое Целое | 1 1 |
| Ингредиенты | Код ингредиента Описание Код продукта Вес продукта | Счетчик Короткий текст Числовой Числовой | 6 символов 20 символов Целое Вещественное | Автогенерация Для блюда 1 70 |
| ИнгредиентыБлюда | Код блюда Код ингредиента | Числовой Числовой | Целое Целое | 1 1 |
| Продукты | Код продукта Название Код категории продукта Калорийность | Счетчик Короткий текст Числовой Числовой | 6 символов 20 символов Целое Целое | Автогенерация Яйцо 1 50 |
| Категории продуктов | Код категории продукта Наименование Описание | Счетчик Короткий текст Короткий текст | 6 символов 20 символов 20 символов | Автогенерация Молочное Из курицы |

Глава 4. Проектирование интерфейса

Основным нефункциональным требованием приложения является удобный для пользователя интерфейс. Далее будет рассмотрен интерфейс и его отдельные формы.

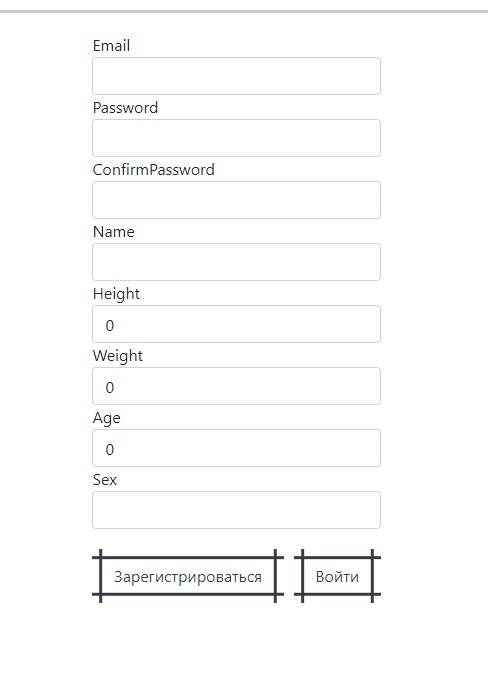
Начальной формой является форма для авторизации. Форма для авторизации представлена на рисунке 4.1.



***Рисунок 4.1. Форма авторизации***

Данная форма предназначена для входа для авторизированных пользователей. Пользователь вводит почту и пароль. После нажатия кнопки Войти, при корректно введенных данных, пользователя перенаправит на основную страницу приложения.

Если пользователь не зарегистрирован, то необходимо нажать на кнопку Регистрация и перейти на страницу регистрации. Форма для регистрации представлена на рисунке 4.2.

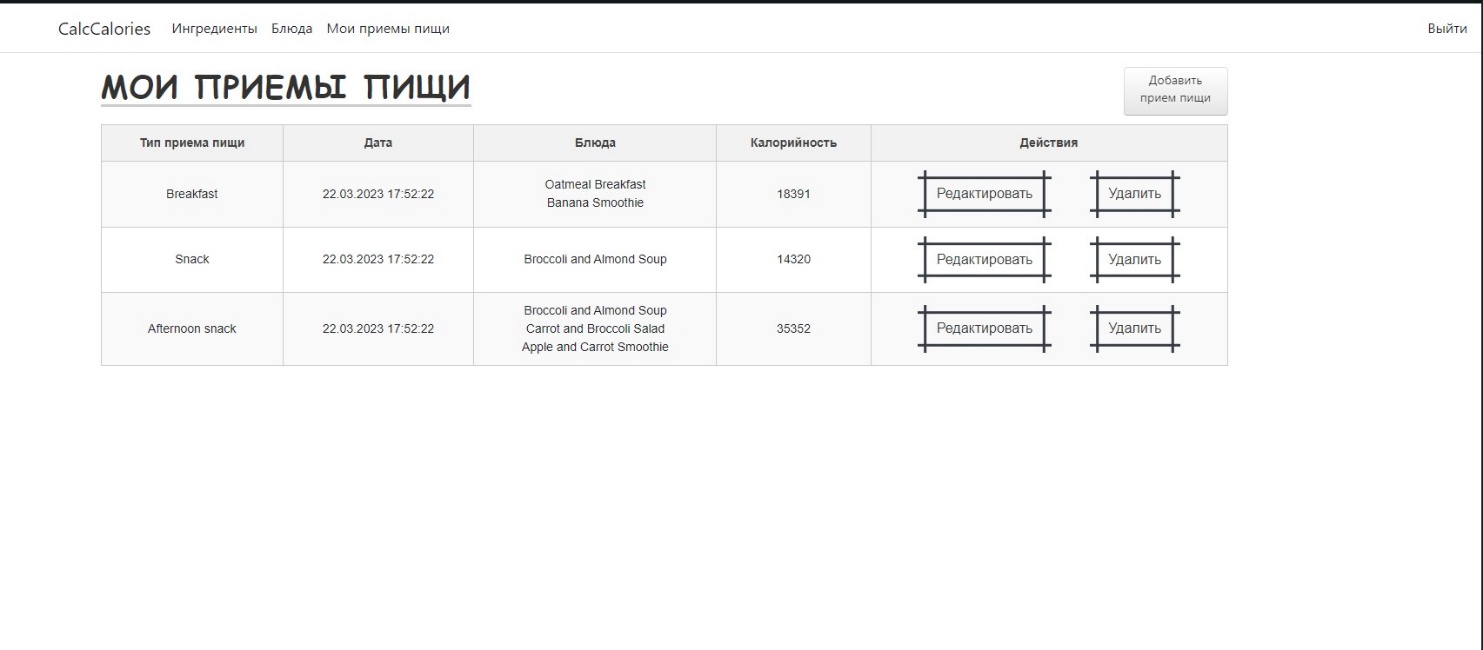


***Рисунок 4.2. Форма регистрации***

Данная форма необходима для регистрации пользователь. Предоставляется возможность ввести личные данные, почту и пароль. После нажатия кнопки Зарегистрироваться, при корректно введенных данных, пользователя перенаправит на основную страницу приложения.

Далее на рисунке 4.3 представлена основная страница приложения для расчета калорийности продуктов и блюд. На данной странице изображены приемы пищи авторизированного пользователя. Про каждый прием пищи известен тип, дата, блюда и общая калорийность приема пищи. Представлены кнопки Редактировать, Удалить для взаимодействия с конкретным приемом пищи. Кнопка Добавить прием пищи для создания нового приема пищи.

На рисунке 4.3 представлена верхняя часть страницы (хедер), на котором присутствуют ссылки на все страницы приложения, такие как Приемы пищи, Ингредиенты, Блюда. Кнопка Выйти позволяет выйти из текущего аккаунта и перейти на страницу авторизации.



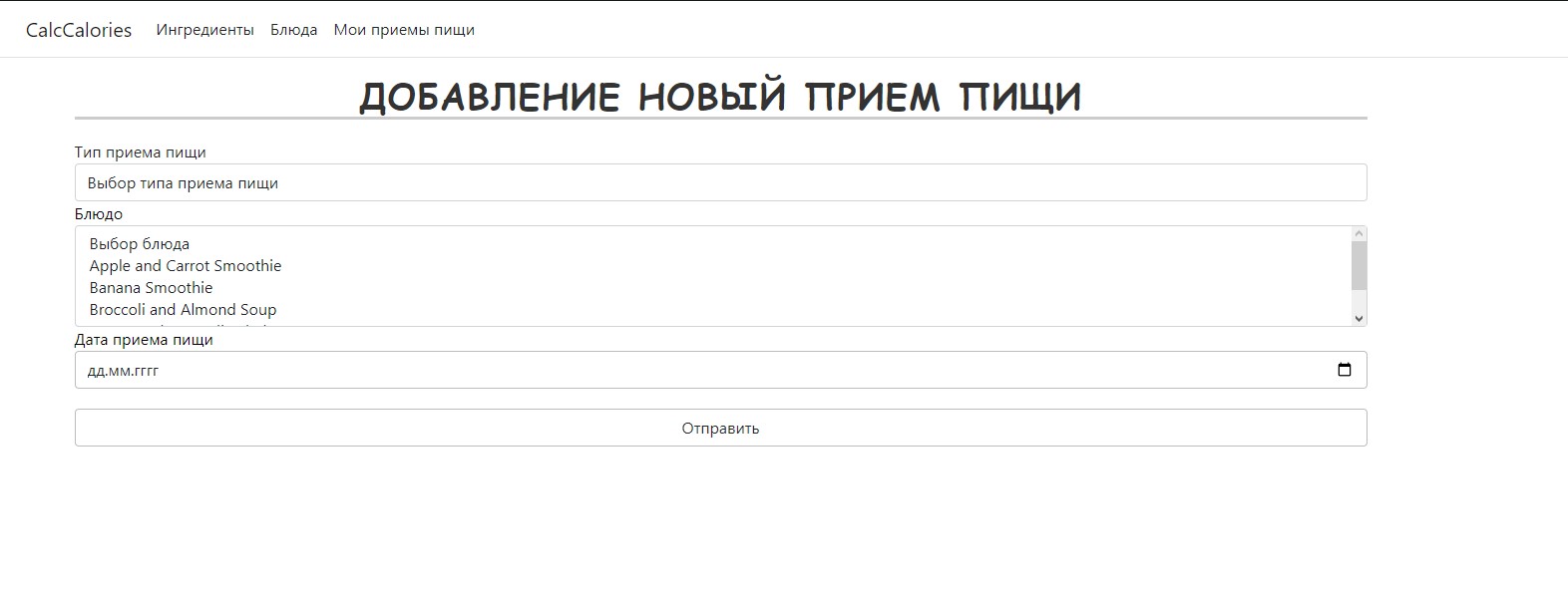
***Рисунок 4.3. Страница приемов пищи.***

При нажатии на кнопку Редактировать происходит переход на страницу Редактирования конкретного приема пищи. Форма редактирования приема пищи представлена на рисунке 4.4.

***Рисунок 4.4. Страница редактирования приема пищи.***

При нажатии на кнопку Удалить происходит удаления элемента и обновление таблицы.

При нажатии на кнопку Добавить прием пищи происходит переход на страницу Создания приема пищи. Форма создания приема пищи представлена на рисунке 4.5.



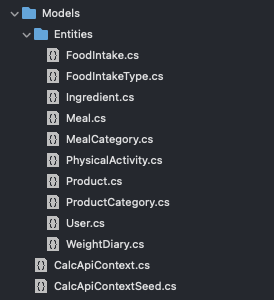
***Рисунок 4.4. Страница редактирования приема пищи.***

Глава 5. Реализация приложения

На этапе реализации приложения разрабатывается программный код. Создается ранее проанализированная и спроектированная база данных, реализуется необходимых функционал и создается спроектированный интерфейс.

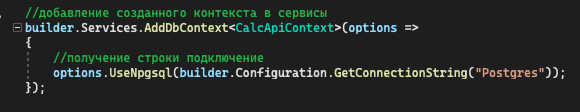
5.1 Описание сущностей

В данном приложение будет использоваться Entity Framework, который делает возможным связывать класс C# и сущностями в базе данных. На рисунке 5.1 представлены классы необходимых сущностей, таких как User, WeightDiary, ProductCategory, Product, PhysicalActivity, Meal, MealCategory, Ingredient, FoodIntake, FoodIntakeType.



***Рисунок 5.1 Структура папки Models.***

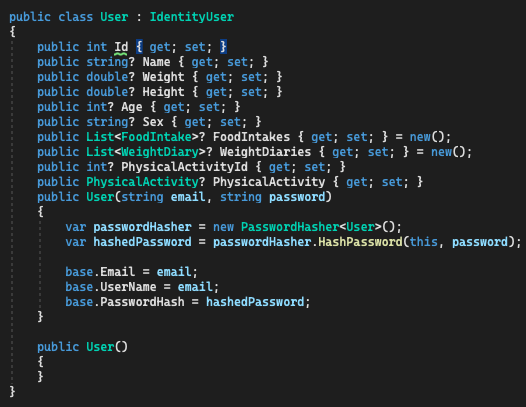
Классы CalcApiContext, который наследуется от класса Microsoft.EntityFrameworkCore.DbContext и создает связи между программными классами и сущностями в базе данных, и CalcApiContextSeed отвечают за взаимодействие с базой данной. В CalcApiContextSeed создаются экземпляры классов из папки Entitites, формируются связи между ними, и база данных заполняется данными экземплярами. Также необходимо подключится к базе данных из файла program.cs, приложение приобретает контекст данных CalcApiContext для связи с базой данных, используя сервис, который интегрирован в коллекцию сервисов приложения с помощью механизма внедрения зависимостей. Данный код изображен на рисунке 5.2.



***Рисунок 5.2 Добавление созданного контекста в сервис приложения.***

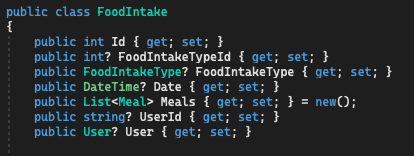
Класс User наследуется от встроенного в пространство имен Microsoft.AspNetCore.Identity.EntityFrameworkCore IdentityUser, который необходим для реализации авторизации и регистрации и имеет поля Email, UserName и PasswordHash.

Также на примере класса User показано создание связей 1 ко многим между сущностями. То есть класс User содержит коллекцию объектов класса FoodIntake, то есть List<FoodIntake>. Код класса User представлен на рисунке 5.3.



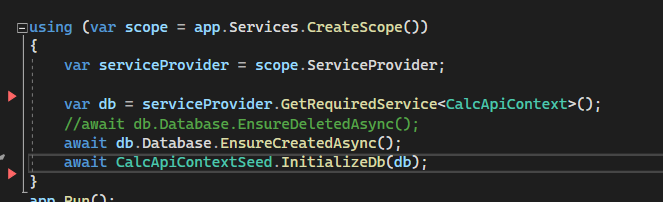
***Рисунок 5.3 Класс User.***

В свою очередь, класс FoodIntake содержит ссылку на экземпляр класса User. Код класса FoodIntake представлен на рисунке 5.4.



***Рисунок 5.4 Класс FoodIntake.***

Создание базы данных и инициализация, то есть заполнение базы данных, осуществляется в коде, представленном на рисунке 5.5.



***Рисунок 5.5 Реализация БД.***

5.2 Реализация регистрации

Интерфейс формы реализован с помощью Razor Pages. Каждое поле для ввода пользовательских данных оснащено tag-хелпером asp-for, которые присваивают данные значения моделям. Модели, в свои очередь, выделены атрибутом BindProperty, который связывает модель и форму ввода, то есть введенные данные присвоятся привязанной модели.

После ввода пользователем необходимых для регистрации данных и нажатия на кнопку Регистрации, выполнится основной метод Регистрации OnPostAsync.

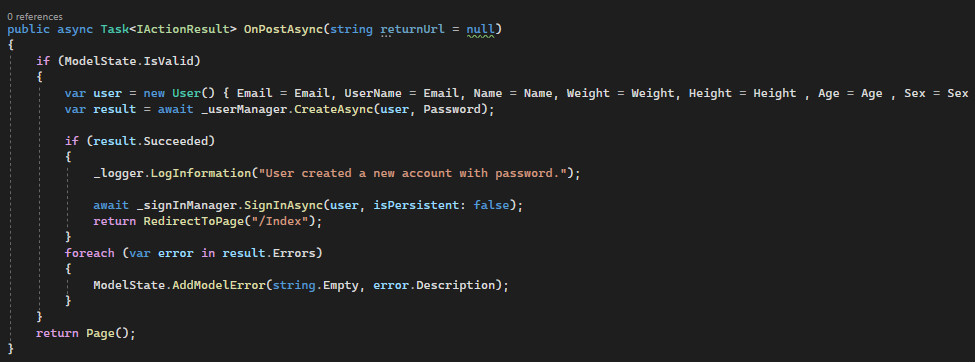
В методе создается новый объект User с заполненными свойствами Email, UserName, Name, Weight, Height, Age, Sex на основе данных, полученных из формы.

Затем метод вызывает метод CreateAsync для объекта UserManager, который создает новую запись пользователя в базе данных на основе переданного объекта User и установленного пароля.

Если создание пользователя прошло успешно (result.Succeeded == true), то метод выполняет вход пользователя в систему с использованием метода SignInAsync и возвращает пользователя на главную страницу.

В противном случае (result.Succeeded == false), метод добавляет ошибки в ModelState, которые затем выводятся на экран пользователю в форме регистрации. В конце метод возвращает страницу регистрации.

Код метода OnPostAsync регистрации представлен на рисунке 5.6.



***Рисунок 5.5 Код метода OnPostAsync регистрации.***

5.3 Реализация авторизации

Интерфейс страницы авторизации реализован аналогично странице регистрации.

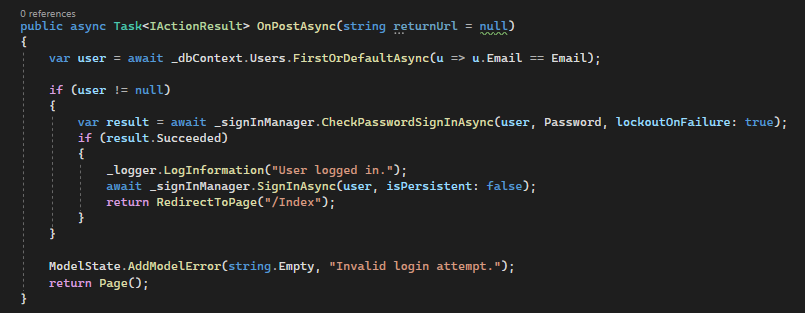
Почта и пароль, введенные пользователем передаются в переменные Email и Password, с помощью атрибута BindProperty. После нажатия на кнопку Войти выполняется основной метод авторизации – OnPostAsync.

В методе выполняется асинхронный запрос к базе данных для поиска пользователя по указанному Email. Если пользователь найден, метод вызывает метод CheckPasswordSignInAsync для объекта SignInManager, который проверяет правильность введенного пароля для найденного пользователя.

Если проверка прошла успешно (result.Succeeded == true), то метод выполняет вход пользователя в систему с использованием метода SignInAsync и возвращает пользователя на главную страницу.

В противном случае (result.Succeeded == false), метод добавляет ошибку в ModelState, которая затем выводится на экран пользователю в форме входа. В конце метод возвращает страницу входа.

Код метода OnPostAsync авторизации представлен на рисунке 5.6.



***Рисунок 5.6 Код метода OnPostAsync авторизации.***

5.4 Реализация страниц

Основная страница, доступная авторизированному пользователю, страница приемов пищи.

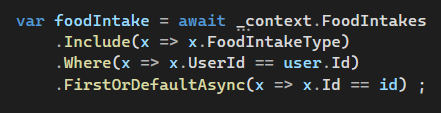
5.4.1 Реализация сервиса Приемов пищи

Для создание более гибкого приложения будет использоваться механизм внедрения зависимостей.

В интерфейсе сервиса приемов пищи будут реализованы следующие методы, запросы к базе данных:

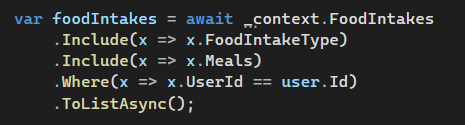
* GetFoodIntakes – получение всех приемов пищи авторизированного пользователя.
* DeleteFoodIntake – удаление конкретного приема пищи.
* GetFoodIntake – получение конкретного приема пищи.

Метод GetFoodIntake получается конкретный прием пищи из базы данных по передаваемому ему индексу. Сначала происходит запрос к текущему авторизованному пользователю, после выполняется запрос к базе данных с учетом необходимых параметров, таких как id приема пищи и id текущего пользователя. Код запроса представлен на рисунке 5.7.



***Рисунок 5.7 Код получения конкретного приема пищи.***

Аналогично выполненный метод GetFoodIntakes выполняется запрос на получение всех приемов пищи у авторизированного пользователя. Код запроса представлен на рисунке 5.8.



***Рисунок 5.8 Код получения всех приемов пищи.***

Метод DeleteFoodIntake, выполяет аналогичный запрос поиска конкретного приема пищи, как и метод GetFoodIntake, с последующим удалением найденного элемента из базы данных.

Коды SQL-запросов, выполняемых во время данных LINQ-запросов представлены в приложении.

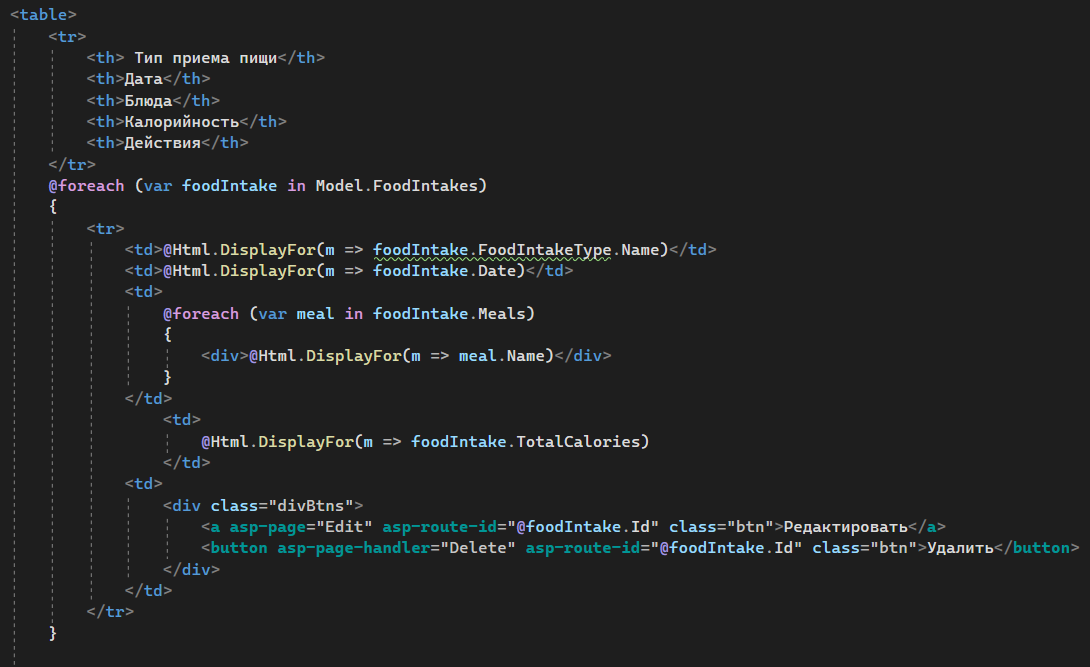
Реализация остальных сервисов выполнена аналогичным образом.

5.4.2 Реализация основной страницы

Она включает в себя таблицу и элементы взаимодействия с таблицей для конкретного пользователя. В данном случае, изображены все приемы пищи пользователя с кнопками для их добавления, удаления и редактирования.

При загрузке страницы срабатывает метод OnGetAsync, который с помощью описанного ранее метода GetFoodIntakes, присваивает свойству модели страницы FoodIntakes полученный лист экземпляров класса FoodIntake.

В HTML-разметке страницы, благодаря возможностям Razor Pages, которые позволяют взаимводействовать с моделями данных, возможно вывести значения свойства FoodIntake, то есть все приемы пищи авторизированного пользователя. Код HTML страницы представлен на рисунке 5.9.

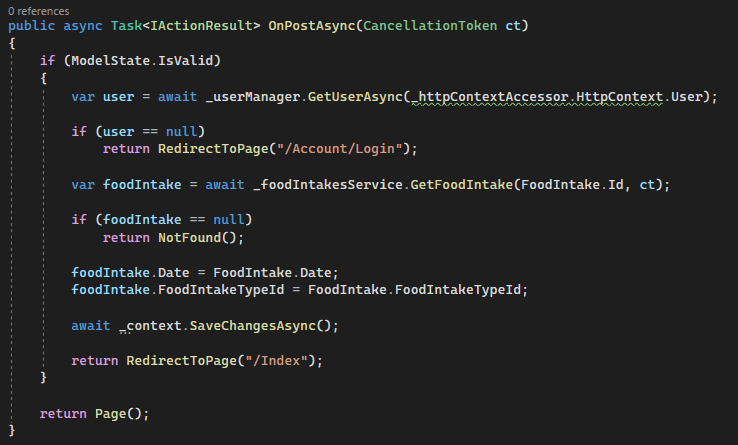


***Рисунок 5.9 HTML код таблицы приемов пищи.***

Нажатие на кнопку Удалить вызывает метод OnPostDelete, в который передается идентификатор текущего объекта. В данном методе вызывается ранее описанный метод DeleteFoodIntake, который принимает идентификатор и совершает удаление данного объекта из базы данных. После чего происходит переадресация на текущую страницу, то есть таблица обновляется без перезагрузки страницы.

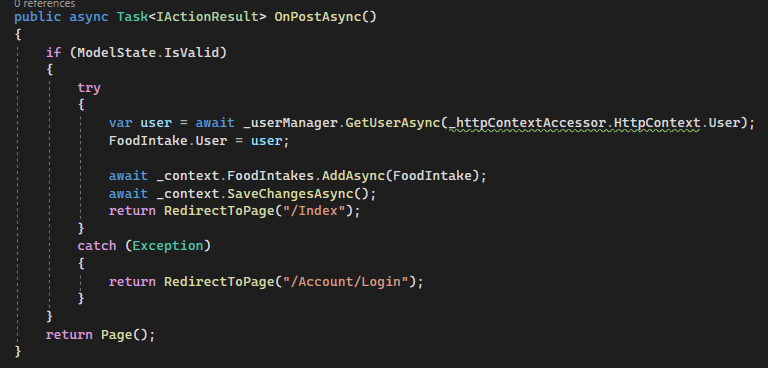
Нажатие на кнопку Редактировать перенаправит пользователя на страницу редактирования. При загрузке странице выполнится метод OnGetAsync, который в свою очередь вызовет метод GetFoodIntake, описанный ранее, и присвоит полученное значение в свойство модели FoodIntake, то есть в полях для редактирования конкретного экземпляра будут выведены текущие данные полей.

После изменения пользователем данных и нажатия на кнопку Сохранить будет вызван метод OnPostAsync. В данном методе перезапишутся поля объекта текущего приема пищи и произойдет сохранение изменений базы данных с помощью метода контекста данных SaveChangesAsync. Код данного метода представлен на рисунке 5.10.



***Рисунок 5.10 Код сохранения изменений приема пищи.***

Также на странице есть кнопка Добавить прием пищи, которая отвечает за создание нового приема пищи. При нажатии на кнопку откроется страница создания приема пищи. HTML-разметка данной страницы содержит теги input, которые с помощью атрибута BindProperty привязаны к свойствам модели страницы. После ввода данных и нажатия на кнопку Добавить срабатывает метод OnPostAsync страницы создания приема пищи. В данном методе благодаря BindProperty имеется созданный объект класса FoodIntake, ему присваивается текущий авторизованный пользователь в своей FoodIntake.User, после чего с помощью метода контекста данных AddAsync новый созданный объект сохраняется в базе данных. После открывается основная страница, на которой изображен новый созданный объект. Код метода OnPostAsync изображен на рисунке 5.11.



***Рисунок 5.11 Код создания приема пищи.***

Остальные страницы с функционалом разработаны аналогичным образом. Код всего приложения представлен ссылкой на репозиторий Git в Приложении.

5.5 Верстка страниц

Верстка приложения — это процесс создания дизайна и разметки веб-страниц, включая элементы интерфейса, структуру и оформление визуального контента, такого как текст, изображения, анимации и другие элементы.

Верстка приложений важна, потому что она позволяет создавать качественный пользовательский интерфейс, улучшающий удобство и доступность использования веб-страниц.

Для создания удобного интерфейса был использован чистый css для создания собственных стилей и css-фреймворк bootstrap.

Заключение

Во время разработки приложения для расчета калорийности продуктов и блюд были получены некоторые результаты, такие как:

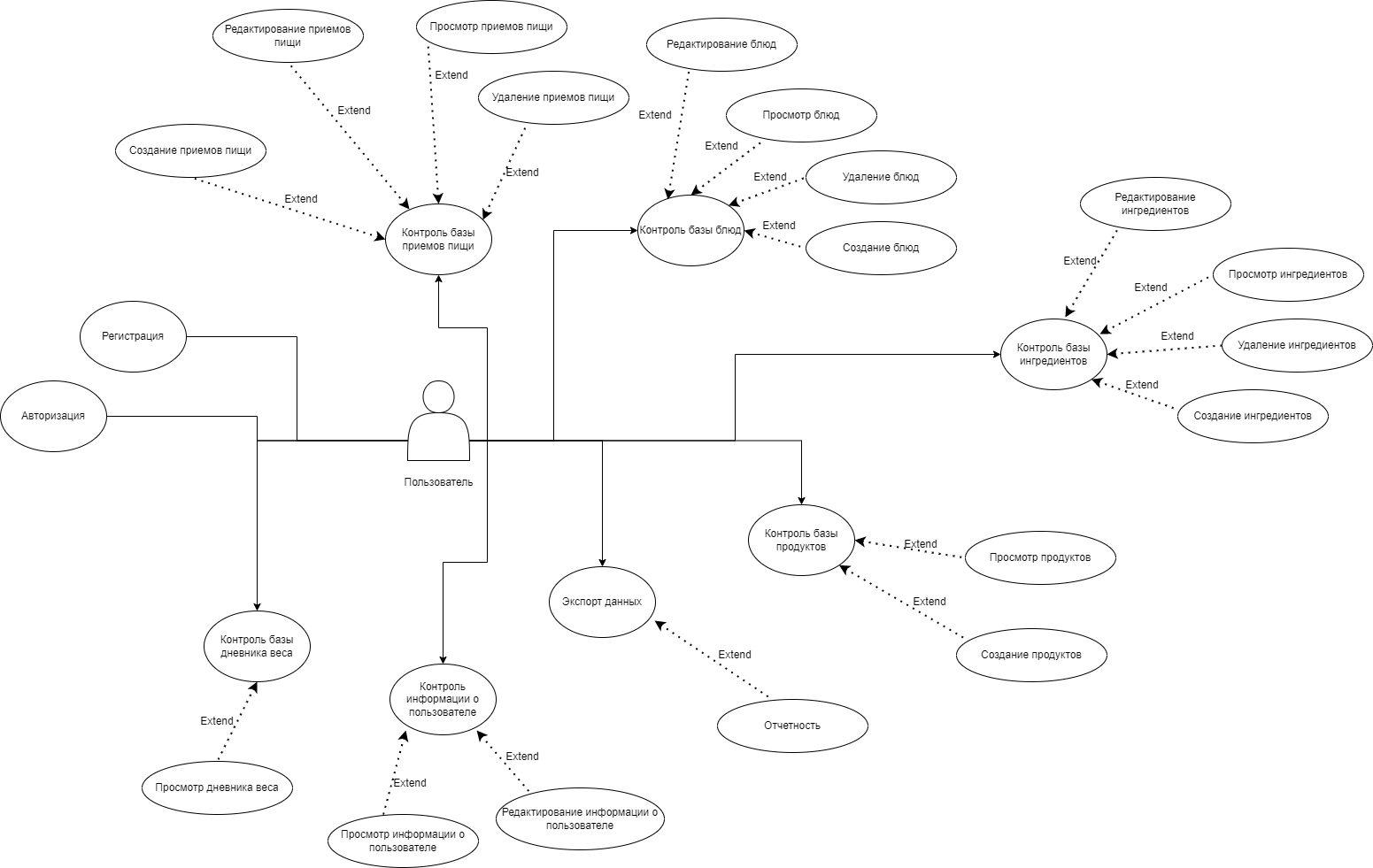
* Анализирование предметной области “Расчет калорийности продуктов и блюд”.
* Были выявлены и разобраны аналоги данной системы, функциональные и нефункциональные требования к приложению.
* Проанализированы бизнес-процессы и прецеденты предметной области.
* Спроектирована база данных. Совершен процесс нормализации базы данных.
* Спроектирован пользовательский интерфейс.
* Реализовано приложение.

Результатом данной курсовой работы является приложение для расчета калорийности продуктов и блюд. Разработка данного приложения для отслеживания калорий представляет собой ценный вклад в область здоровья и благополучия. Предоставляя пользователям инструменты, необходимые для контроля за потреблением пищи и принятия обоснованных решений относительно своего рациона, это приложение способно содействовать здоровому образу жизни и улучшению общего благосостояния.

Данное приложение имеет безграничный потенциал к развитию и совершенству. Например, кроссплатформенность, чтобы любой пользователь в любой момент мог воспользоваться данным приложением, добавление полезных рецептов, чтобы пользователи могли добавлять и просматривать различные рецепты, способствующие здоровому образу жизни.

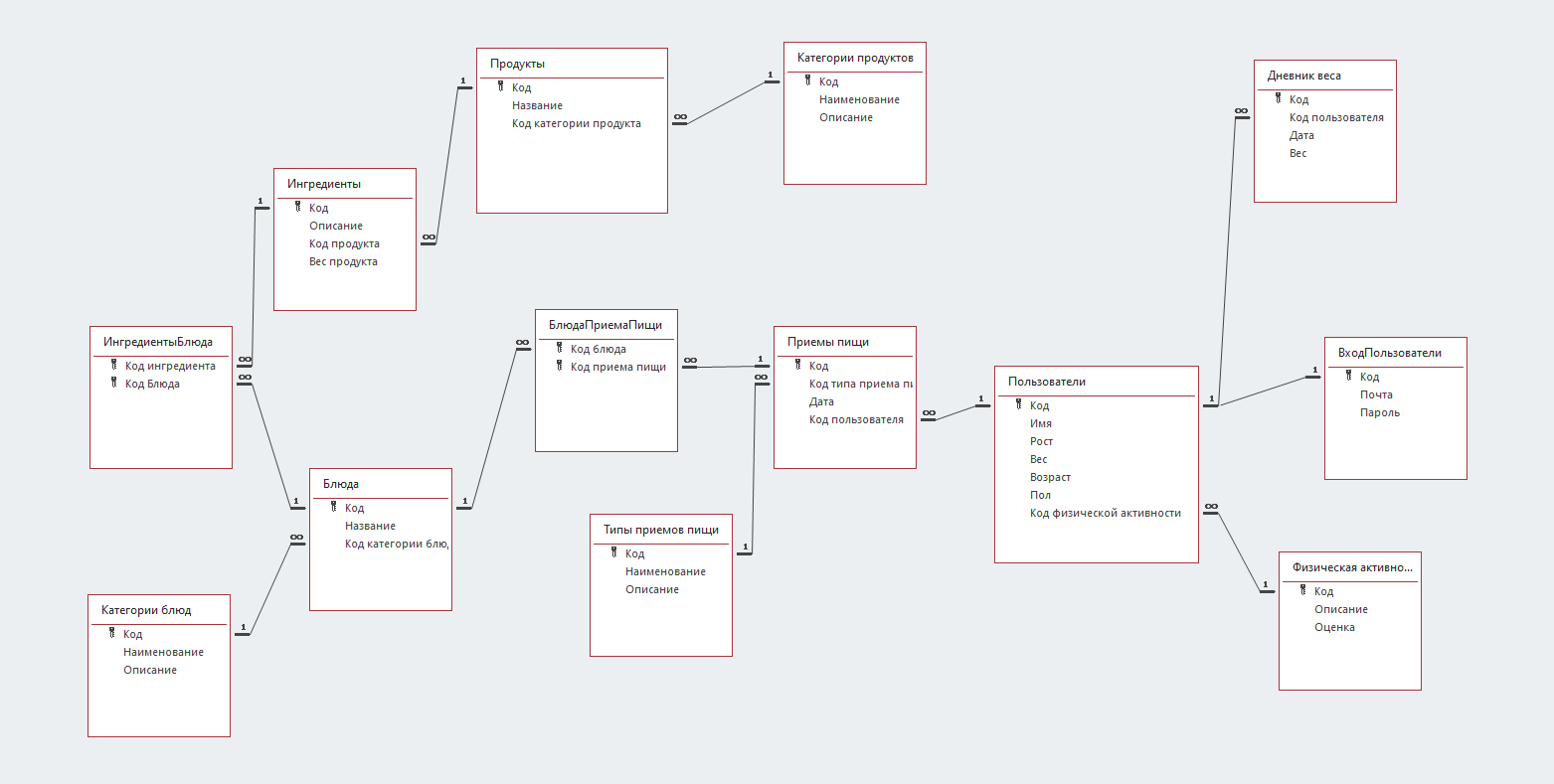
Библиографический список

Приложение А. Диаграмма прецедентов



***Рисунок А.1. Диаграмма прецедентов для приложения для расчета калорийности продуктов и блюд***

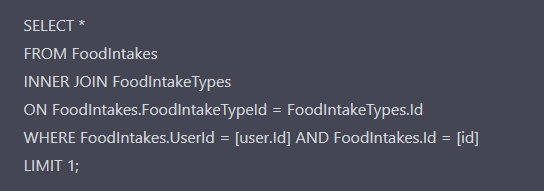
Приложение Б. ER-диаграмма базы данных



***Рисунок Б.1. Схема базы данных приложения для расчета калорийности продуктов и блюд***

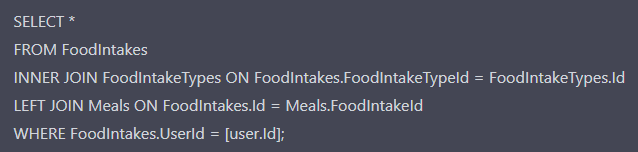
Приложение C. SQL-запросы

SQL-запрос получения конкретного Приема пищи представлен на рисунке C.1.



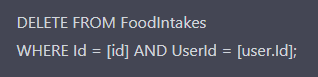
***Рисунок C.1. Запрос на получение приема пищи***

SQL-запрос получения всех приемов пищи конкретного пользователя представлен на рисунке C.2.



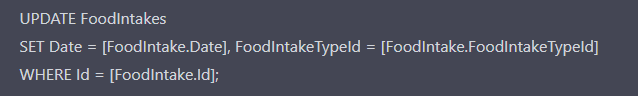
***Рисунок C.2. Запрос на получение приемов пищи***

SQL-запрос на удаление конкретного приема пищи представлен на рисунке C.3.



***Рисунок C.3. Запрос на удаление приема пищи***

SQL-запрос на редактирование приема пищи представлен на рисунке C.4.



***Рисунок C.4. Запрос на редактирование приема пищи***

Приложение D. Код программы

Ссылка на репозиторий Git - https://github.com/EgorBelov/Calc.