Пермский филиал федерального государственного автономного образовательного учреждения высшего образования «Национальный исследовательский университет

«Высшая школа экономики»

Факультет социально-экономических и компьютерных наук

Образовательная программа бакалавриата «Программная инженерия»

**ОТЧЕТ  
по практике**

Вид практики: Профессиональный

Тип практики: Учебная практика

Выполнил студент группы ПИ-21-1

Белов Егор Александрович

(Фамилия, Имя, Отчество)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись)

Проверил:

Руководитель практики

Доцент кафедры информационных технологий в бизнесе

(должность ученая степень, ученое звание)

Замятина Елена Борисовна

(Фамилия, Имя, Отчество)

\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(оценка) (подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата)

**Пермь, 2023**

**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 65 страницу и 7 глав. В данной работе содержится 25 рисунков, 8 приложений.

Оглавление

[Введение 5](#_Toc136625953)

[Глава 1. Анализ предметной области 7](#_Toc136625954)

[1.1 Описание существующих решений 7](#_Toc136625955)

[1.1.1 MyFitnessPal 7](#_Toc136625956)

[1.1.2 Yazio 8](#_Toc136625957)

[1.1.3 FatSecret 10](#_Toc136625958)

[1.1.4 Онлайн сервисы 10](#_Toc136625959)

[1.1.5 Итоги 11](#_Toc136625960)

[1.2 Назначение системы и основные характеристики 12](#_Toc136625961)

[1.3 Описание бизнес-процессов 13](#_Toc136625962)

[1.4 Описание прецедентов 13](#_Toc136625963)

[1.5 Требования к системе 17](#_Toc136625964)

[Глава 2. Технологии и выбор средств реализации приложения 19](#_Toc136625965)

[2.1 Выбор СУБД для базы данных 19](#_Toc136625966)

[2.2 Выбор программных средств реализации 20](#_Toc136625967)

[Глава 3. Проектирование Базы данных 21](#_Toc136625968)

[3.1 Нормализация 21](#_Toc136625969)

[3.1.1 Функциональные зависимости 21](#_Toc136625970)

[3.1.2 Приведение в 1НФ 21](#_Toc136625971)

[3.1.3 Приведение в 2 НФ 23](#_Toc136625972)

[3.1.4 Приведение в 3 НФ 25](#_Toc136625973)

[3.2 Описание таблиц и их составляющих 25](#_Toc136625974)

[Глава 4. Проектирование интерфейса 28](#_Toc136625975)

[Глава 5. Реализация приложения 32](#_Toc136625976)

[5.1 Описание сущностей 32](#_Toc136625977)

[5.2 Реализация регистрации 34](#_Toc136625978)

[5.3 Реализация авторизации 35](#_Toc136625979)

[5.4 Реализация страниц 36](#_Toc136625980)

[5.4.1 Реализация сервиса Приемов пищи 36](#_Toc136625981)

[5.4.2 Реализация основной страницы 37](#_Toc136625982)

[5.5 Реализация поиска 40](#_Toc136625983)

[5.6 Экспорт таблиц 41](#_Toc136625984)

[5.7 Верстка страниц 42](#_Toc136625985)

[Глава 6. Тестирование 44](#_Toc136625986)

[Глава 7. Развертывание 46](#_Toc136625987)

[Заключение 50](#_Toc136625988)

[Библиографический список 51](#_Toc136625989)

[Приложение A. Диаграммы активностей 52](#_Toc136625990)

[Приложение Б. Диаграмма прецедентов 55](#_Toc136625991)

[Приложение В. ER-диаграмма базы данных 56](#_Toc136625992)

[Приложение Г. Диаграмма классов приложения 57](#_Toc136625993)

[Приложение Д. SQL-запросы 58](#_Toc136625994)

[Приложение Е. Код программы 59](#_Toc136625995)

[Приложение Ж. Техническое задание 60](#_Toc136625996)

[Приложение И. Руководство пользователя 63](#_Toc136625997)

Введение

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 вести 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 информация о здоровом образе жизни, а более точно, информация о продуктах, блюдах и их калорийности.

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

Целью 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. Описание существующих решений

В данном разделе будут рассмотрены существующие и доступные для анализа 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.

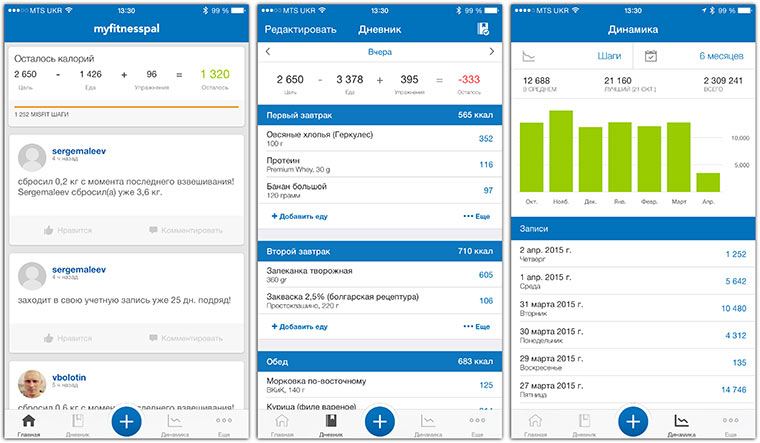
Преимущества:

* 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.

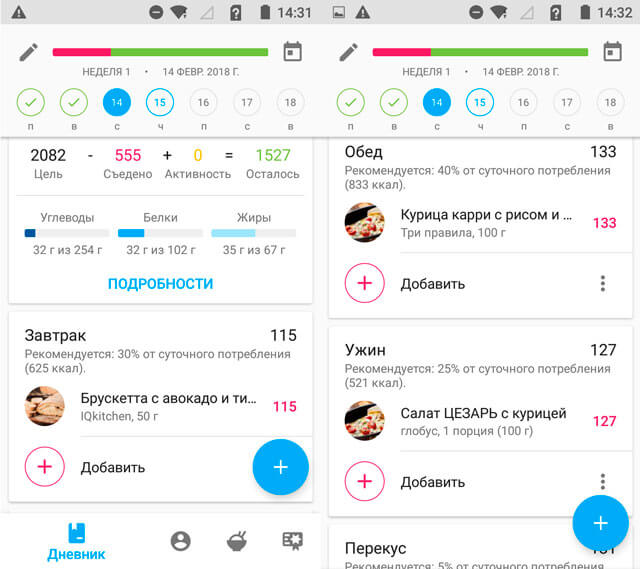
Преимущества:

* 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***

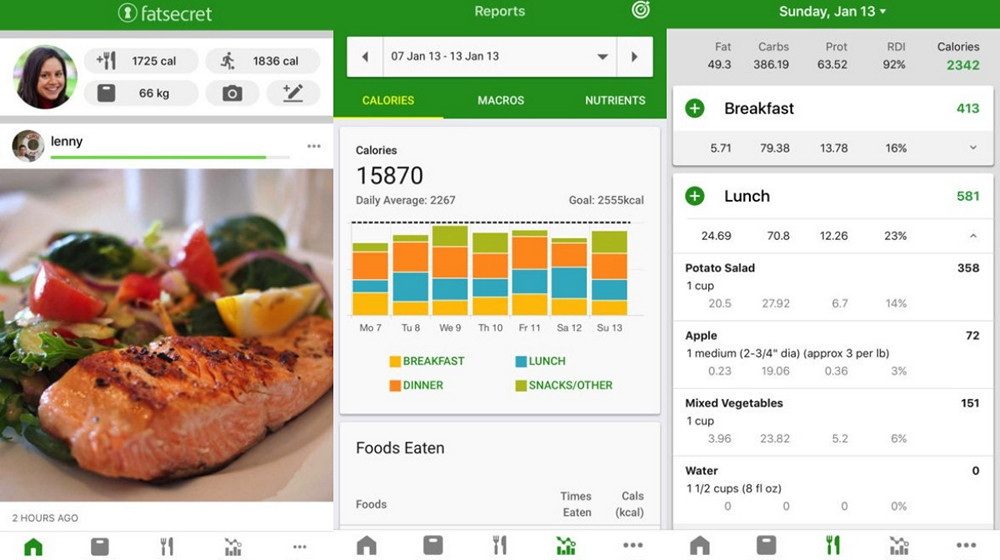
В целом, Yazio может быть полезным инструментом для пользователей, которые хотят отслеживать потребление калорий и прогресс в потере веса, но оно может не подойти всем из-за своих ограничений и неточной информации.

* + 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

Необходимость разработки нового приложения для расчета калорийности продуктов и блюд остается высокой, несмотря на существующие аналоги, поскольку они имеют некоторые недостатки. Одним из таких недостатков является наличие платной подписки для расширения функционала, что может быть неприемлемо для пользователей, которые ищут бесплатное решение. Кроме того, многие из существующих приложений имеют неточности в объемной базе данных, что может привести к неточным результатам при расчете калорийности блюд.

Также, не все существующие приложения имеют возможность индивидуальной настройки целей похудения или достижения определенных показателей. Это может быть проблемой для тех пользователей, которые имеют уникальные потребности и требуют индивидуального подхода.

Некоторые приложения также могут быть не мультиплатформенными, что ограничивает их использование только на определенных устройствах и может быть неудобным для пользователей, которые предпочитают использовать разные устройства в разное время.

Таким образом, разработка нового приложения для расчета калорийности продуктов и блюд имеет необходимость для того, чтобы обеспечить пользователям более точный и удобный инструмент, который учитывает их уникальные потребности и предоставляет доступ к расширенным функциям без дополнительных затрат. Новое приложение может также устранить недостатки существующих аналогов, обеспечивая более точную базу данных продуктов и блюд, индивидуальную настройку целей и мультиплатформенность.

* 1. Назначение системы и основные характеристики

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

Каждый пользователь приложения – зарегистрированный пользователь, который может добавлять записи в базу данных о приемах 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 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.

Диаграммы активностей бизнес-процессов представлены в Приложении А.

* 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. Диаграмма прецедентов представлена в Приложении Б.

***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 |
| Триггер | 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.

Список нефункциональных требований:

* Разработка проекта в среде программирования Visual Studio.
* Использование языка программирования С# и фреймворка ASP.NET.
* Взаимодействие с базой данных будет осуществляться при помощи PG Admin4 и Entity Framework Core.
* Ограничение на ввод посторонних символов будет установлено.
* Пользователи будут получать сообщения об ошибке в случае исключительных ситуаций.
* Приложение будет обладать удобным пользовательским 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: Информация о пользователе должна включать его Имя, Пол, Возраст, Вес, Рост, Почту, Пароль и Уровень физической активности.

2 ФЗ: Про дневник веса есть информация об его Пользователя, Дату, Вес.

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

4 ФЗ: В информации о приеме пищи можно однозначно определить Тип приема пищи, Дату, Принимаемое блюдо, Пользователя.

5 ФЗ: Тип приема пищи включает в себя Название типа, Описание.

6 ФЗ: Информация о блюде включает в себя Название и Категорию.

7 ФЗ: Информация о категории блюда включает в себя Название категории и Описание.

8 ФЗ: По продукту можно однозначно определить Название продукта, Категорию и Калорийность.

9 ФЗ: Информация о категории продукта включает в себя Наименование категории и Описание.

10 ФЗ: Информация об ингредиенте включает в себя 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. Имя пользователя.
2. Рост пользователя.
3. Вес пользователя.
4. Возраст пользователя.
5. Пол пользователя.
6. Почта пользователя.
7. Пароль пользователя.
8. Описание физической активности.
9. Оценка физической активности.
10. Дата измерения веса.
11. Вес.
12. Дата и время приема пищи.
13. Наименование типа приема пищи.
14. Описание типа приема пищи.
15. Название блюда.
16. Описание категории блюда.
17. Наименование категории блюда.
18. Название продукта.
19. Калорийность продукта.
20. Код категории продукта.
21. Наименование категории продукта.
22. Описание категории продукта.
23. Вес продукта.

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.

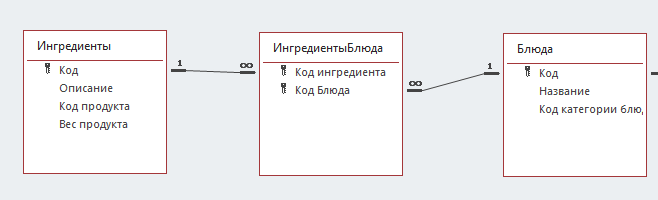
“Продукты”, которое включает атрибуты: Код продукта, Название, Код категории продукта, Калорийность.

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

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

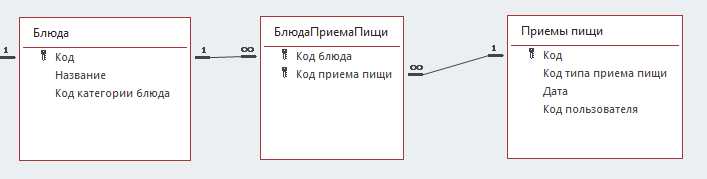
В приведённых отношениях отсутствуют частичные функциональные зависимости.

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 В.

3.2 Описание таблиц и их составляющих

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

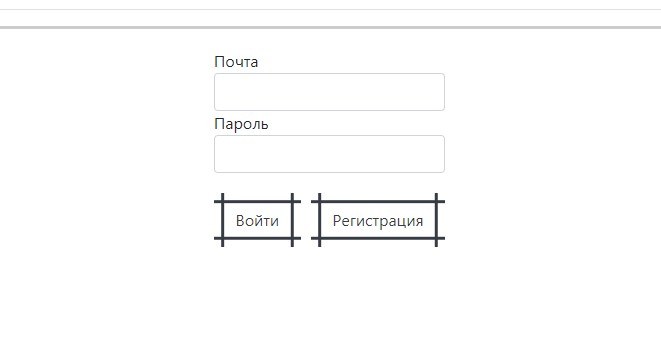
***Таблица 3.1 Физическая модель данных***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 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 |
| 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 |
| Объект | Атрибут | Вид атрибута | Свойства | Пример |
| Категории блюда | Код категории блюда Наименование Описание | Счетчик Короткий текст Короткий текст | 6 символов 20 символов 20 символов | Автогенерация Жареное Пожарили |
| 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.

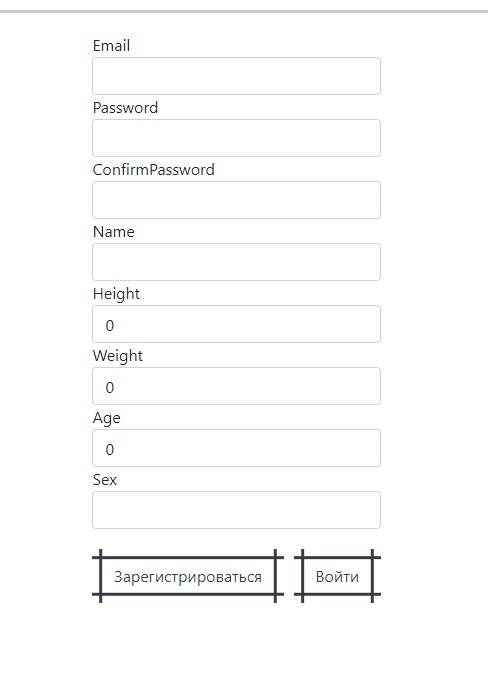
Начальной формой является форма для авторизации. Форма для авторизации представлена на рисунке 4.1.



***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.

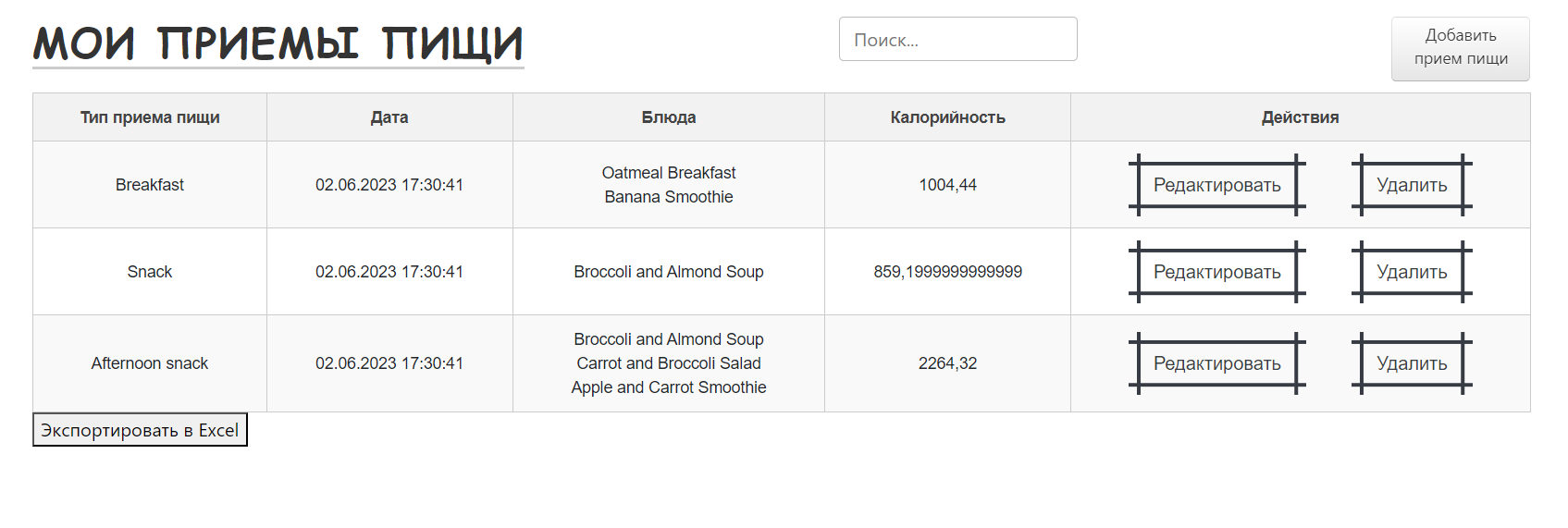


***Рисунок 4.2. Форма регистрации***

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.

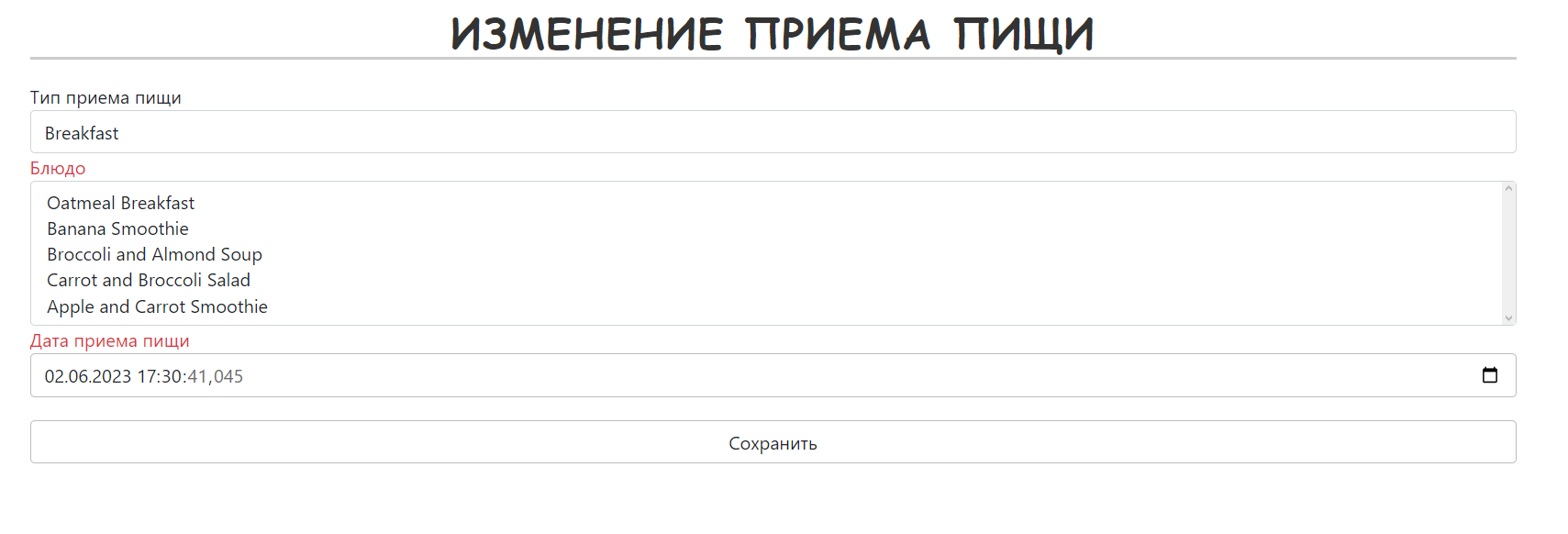
Далее на рисунке 4.3 представлена основная страница приложения для расчета калорийности продуктов и блюд. На данной странице изображены приемы пищи авторизированного пользователя. Про каждый прием пищи известен тип, дата, блюда и общая калорийность приема пищи. Представлены кнопки Редактировать, Удалить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

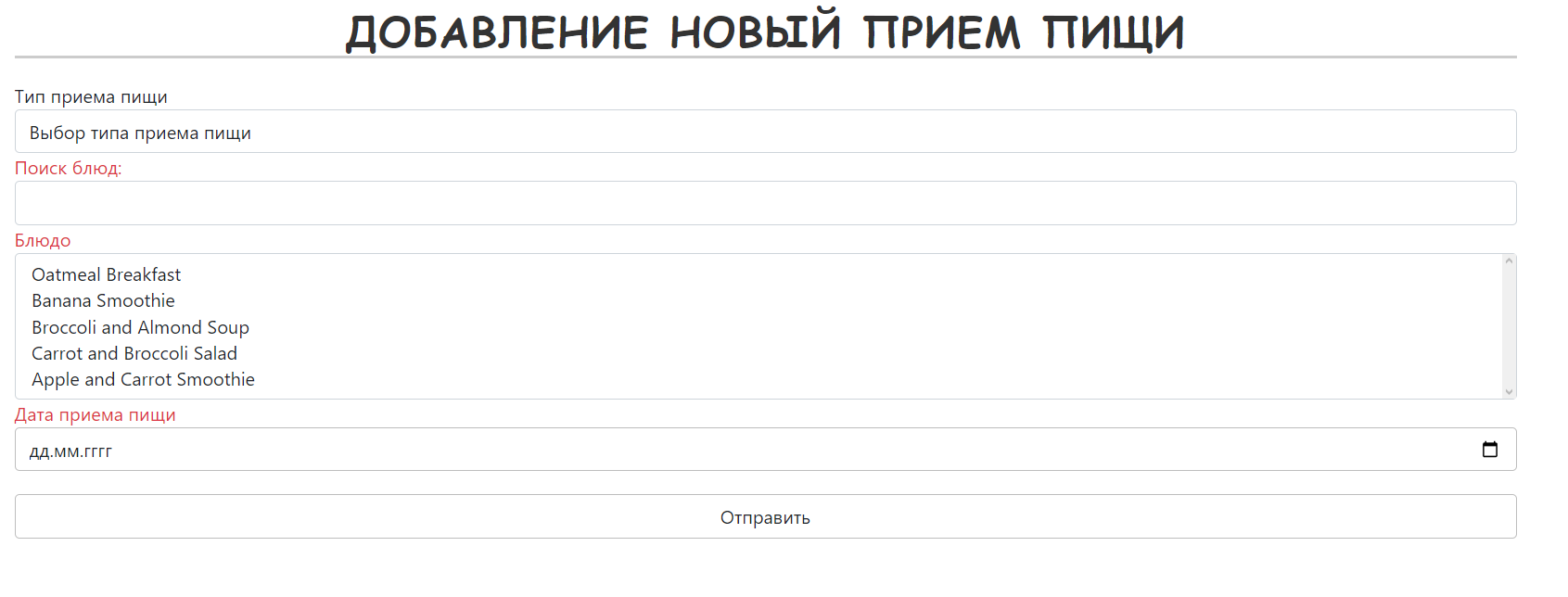
При нажатии на кнопку Редактировать происходит переход на страницу Редактирования конкретного приема пищи. Форма редактирования приема пищи представлена на рисунке 4.4.



***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.



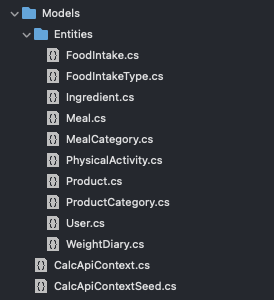
***Рисунок 4.5. Страница создания приема пищи.***

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.

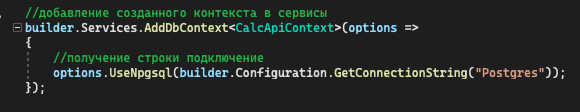
5.1 Описание сущностей

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. Диаграмма классов приложения представлена в Приложении Г.



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

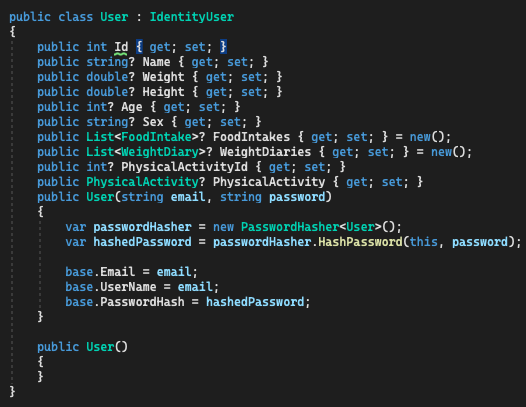
Классы CalcApiContext, который наследуется от класса Microsoft.EntityFrameworkCore.DbContext и создает связи между программными классами и сущностями в базе данных, и CalcApiContextSeed отвечают за взаимодействие с базой данной. В CalcApiContextSeed создаются экземпляры классов из папки Entitites, формируются 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.***

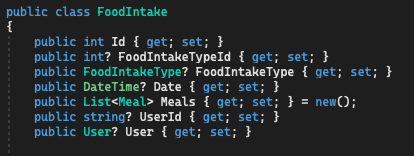
Класс User наследуется от встроенного в пространство имен Microsoft.AspNetCore.Identity.EntityFrameworkCore IdentityUser, который необходим для реализации авторизации и регистрации и имеет поля Email, UserName и PasswordHash.

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.



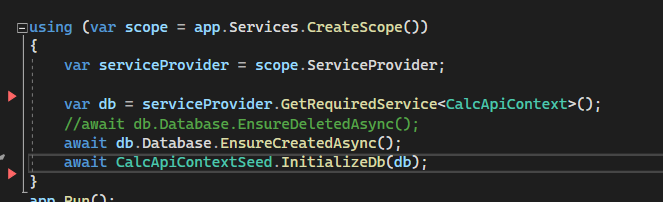
***Рисунок 5.3 Класс User.***

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.

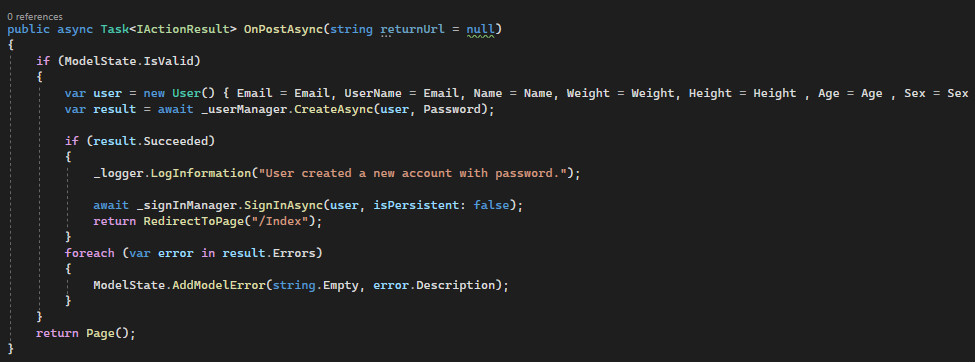
В методе создается новый объект User с заполненными свойствами Email, UserName, Name, Weight, Height, Age, Sex на основе данных, полученных из формы.

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.

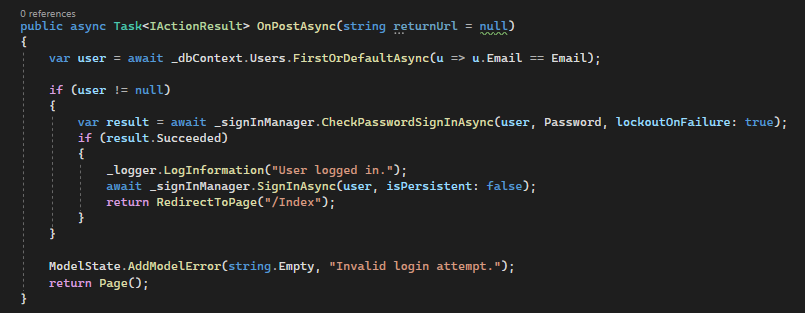
Почта и пароль, введенные пользователем передаются в переменные Email и Password, с помощью атрибута BindProperty. После нажатия на кнопку Войти выполняется основной метод авторизации – OnPostAsync.

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.

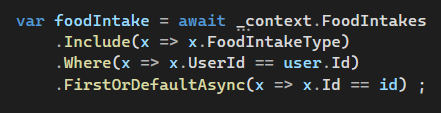
5.4.1 Реализация сервиса Приемов пищи

Для создание более гибкого приложения будет использоваться механизм внедрения зависимостей.

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

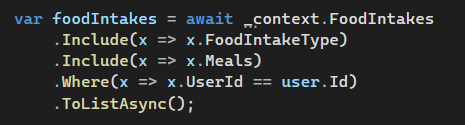
* GetFoodIntakes – получение всех приемов пищи авторизированного пользователя.
* DeleteFoodIntake – удаление конкретного приема пищи.
* GetFoodIntake – получение конкретного приема пищи.

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, выполняет 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 Д.

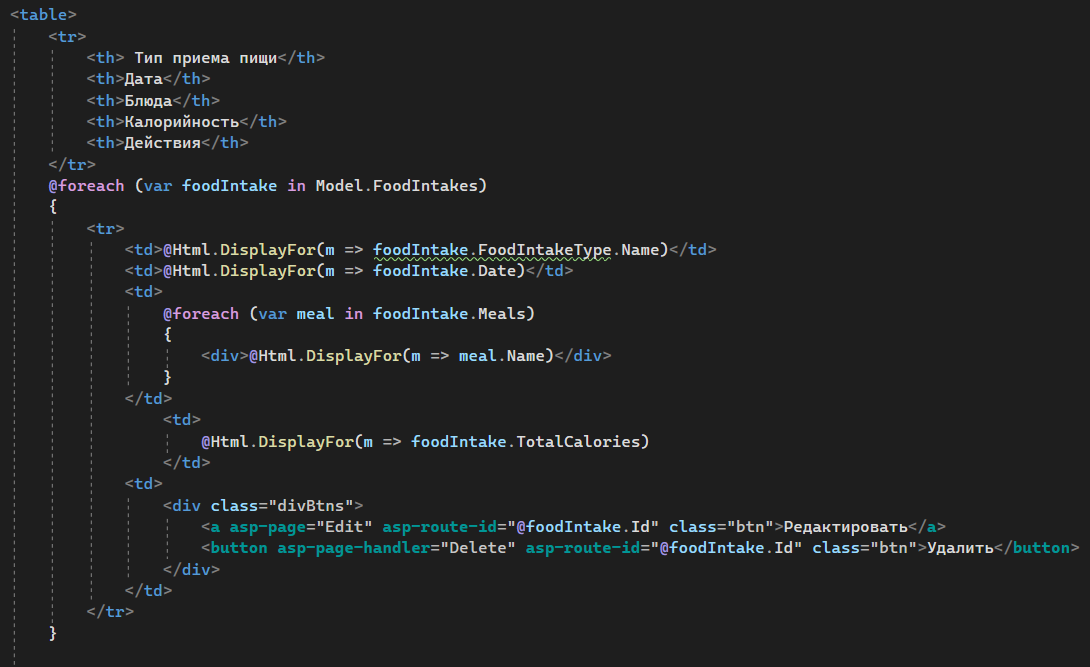
Реализация остальных сервисов выполнена аналогичным образом.

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.

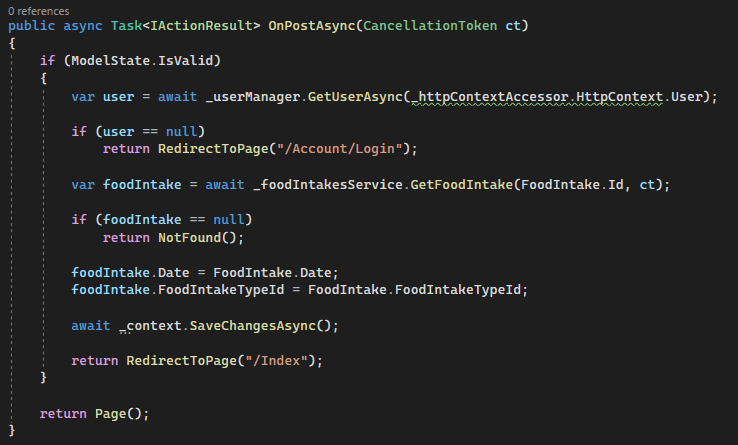


***Рисунок 5.9 HTML код таблицы приемов пищи.***

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.

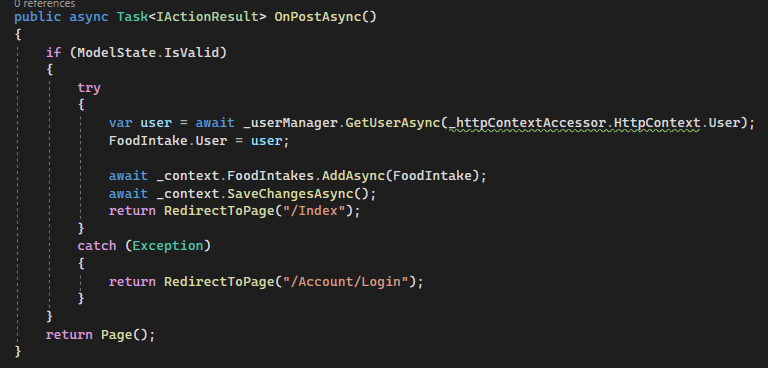
Нажатие на кнопку Редактировать перенаправит пользователя на страницу редактирования. При загрузке странице выполнится метод OnGetAsync, который в свою очередь вызовет метод GetFoodIntake, описанный ранее, и присвоит полученное значение в свойство модели FoodIntake, то есть в полях для редактирования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

5.5 Реализация поиска

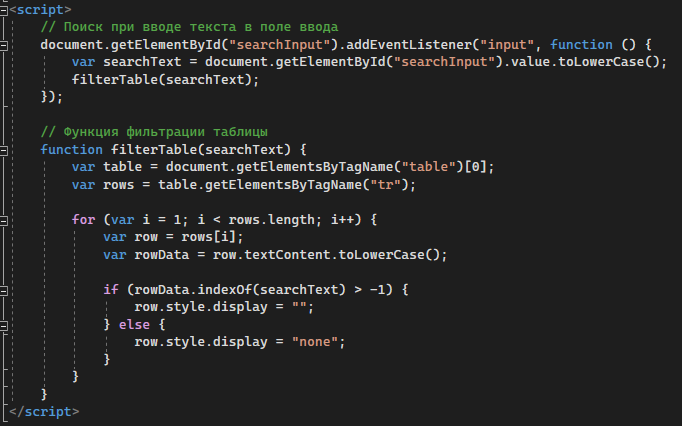
Для реализации функциональности поиска был выбрал язык программирования JavaScript благодаря его способности обеспечивать интерактивность в реальном времени, легкой интеграции с другими веб-технологиями и широкой поддержке браузерами.

Представленный на рисунке 5.12 фрагмент кода иллюстрирует реализацию функции поиска на веб-странице. Функция поиска позволяет пользователям вводить запрос в поле ввода, а затем фильтровать таблицу на основе введенного текста. Когда пользователь вводит текст в поле поиска, активируется слушатель события "ввод", запускающий функцию filterTable для выполнения операции фильтрации.

В функции filterTable получают элемент таблицы и содержащиеся в нем строки. Перебирая каждую строку, начиная с индекса 1 (исключая строку заголовка таблицы), функция получает текстовое содержимое строки и преобразует его в нижний регистр для сравнения без учета регистра.

Сравнение поискового запроса (хранящегося в переменной searchText) и содержимого строки производится с помощью метода indexOf. Если поисковый запрос найден в содержимом строки, стиль отображения строки устанавливается на значение по умолчанию, что означает ее видимость. И наоборот, если поисковый запрос не найден, стиль отображения строки устанавливается на "none", делая ее скрытой.

В целом, данная реализация обеспечивает простую и эффективную функциональность поиска для таблицы веб-страницы. Пользователи могут искать определенную информацию в таблице, вводя ключевые слова, в результате чего динамически отображаются только те строки, которые соответствуют критериям поиска. Представленный код демонстрирует поиск на странице приемов пищи, на остальных страницах метод реализован аналогичным образом.



***Рисунок 5.12 Код поиска.***

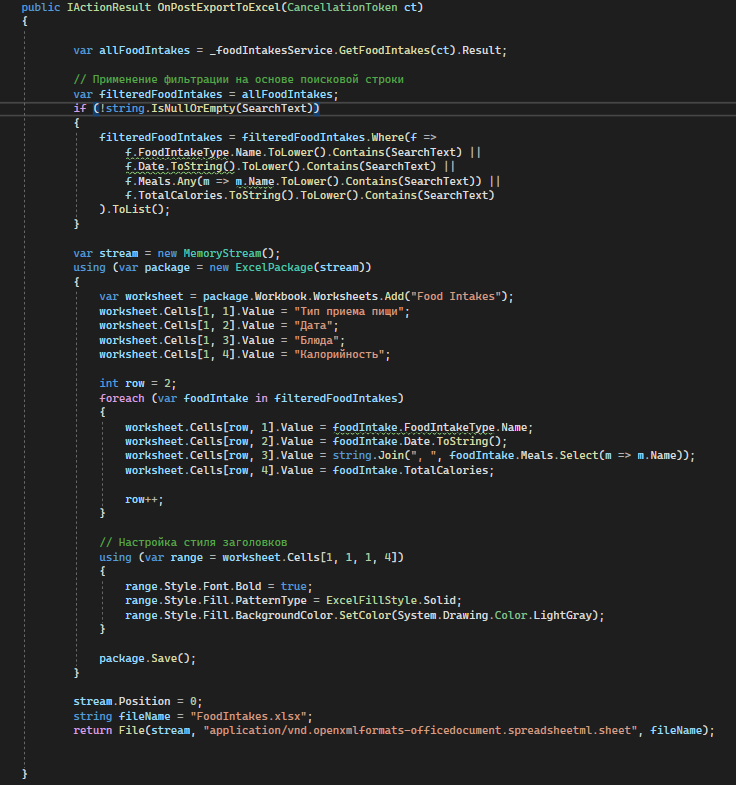
5.6 Экспорт таблиц

На рисунке 5.13 изображен фрагмент кода, демонстрирующий экспорт данных таблицы в формат файла Excel. При нажатии на кнопку “Экспортировать в Excel” вызывается функция ExportToExcel. Данный метод извлекает все приемы пищи из услуги. Кроме того, применяется фильтрацию на основе строки поиска, если таковая предусмотрена. Отфильтрованные данные о приеме пищи затем используются для заполнения рабочего листа Excel.

Для создания файла Excel инициализируется поток памяти, и на основе этого потока создается экземпляр класса ExcelPackage. В пакет добавляется рабочий лист с именем "FoodIntakes” и устанавливаются заголовки для рабочего листа.

Перебирая отфильтрованные приемы пищи, программа заполняет ячейки рабочего листа соответствующими значениями. Информация о каждом приеме пищи, такая как тип приема, дата, порции и общее количество калорий, вставляется в соответствующие ячейки.

После заполнения и стилизации рабочего листа пакет Excel сохраняется, а позиция потока памяти сбрасывается в начало. Код указывает имя файла как "FoodIntakes.xlsx" и возвращает файл в качестве ответа с MIME-типом "application/vnd.openxmlformats-officedocument.spreadsheetml.sheet", позволяя пользователю загрузить его. Представленный код демонстрирует экспорт на странице приемов пищи, на остальных страницах метод реализован аналогичным образом.



***Рисунок 5.13 Код экспорта в excel.***

5.7 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.

Глава 6. Тестирование

Тестирование является важнейшим аспектом разработки программного обеспечения, поскольку оно обеспечивает надежность и корректность реализованных функциональных возможностей. Для данного проекта была выбрана среда тестирования xUnit, популярная среда модульного тестирования для приложений .NET. Цель тестирования - убедиться, что они функционируют так, как ожидается, правильно обрабатывают операции с данными и обеспечивают точные расчеты.

Служба Product была протестирована с помощью среды тестирования xUnit в проекте TestProjectCalcApi. Метод тестирования, TestProductServiceCRUD(), охватывает операции CRUD сервиса. Сначала создается контекст тестовой базы данных с использованием базы данных in-memory. Затем экземпляр класса ProductService инициализируется с тестовым контекстом. Далее тестовый метод выполняет следующие шаги:

* Создание продукта.
* Получение всех продуктов.
* Получение созданного продукта.
* Редактирование продукта.
* Проверка обновленного продукта.
* Удаление продукта.

В целом, этот метод комплексного тестирования проверяет функциональность и целостность службы Product, гарантируя, что продукты могут быть созданы, получены, обновлены и удалены правильно.

Для тестирования было выбрано “безмашинное тестирование”, при котором тестирование проводят живые люди. Для тестирования клиентской части было выбрано ограниченное число человек, студентов первого, второго и третьего курса направления “Программная инженерия” в НИУ ВШЭ Пермь. Само WEB приложение было развернуто на удаленном сервере, к этому приложению получила доступ наша тестовая группа, которая использовала сайт по назначению. После тестирования люди отправляли нам рефлексии по поводу использования приложения, в котором были указаны найденные ошибки, неудобные моменты и пожелания. После анализа всех рефлексий были найдены и исправлены ошибки в клиентской части программы.

При проведении повторных тестов старые ошибки были исправлены, а новые ошибки выявлены не были.

При помощи группы людей удалось относительно быстро проверить несколько сценариев совместимости:

* с различными браузерами (Chrome, Yandex) и различными их версиями;
* что шрифты подобраны корректно.

Также тестеры смогли проверить функциональность приложения и убедиться, что весь функционал приложения работает исправно и соответствует заявленному.

Глава 7. Развертывание

Развертывание приложения на сервере относится к процессу обеспечения доступности приложения для выполнения в серверной среде. В контексте приложений [ASP.NET](https://vk.com/away.php?to=http%3A%2F%2FASP.NET&cc_key=) развертывание включает в себя передачу файлов приложения вместе с любыми необходимыми зависимостями, конфигурациями и ресурсами на веб-сервер, где они могут быть доступны пользователям. Этот процесс гарантирует, что приложение правильно установлено и настроено для эффективной и безопасной работы.

Наиболее распространенные типы развертывания для приложений ASP.NET включают:

* Развертывание на файловой системе.
* Пакет веб-развертывания.
* Web Deploy.
* Azure App Service.
* Контейнеры Docker.
* Веб-развертывание IIS.

Развертывание на файловой системе - это тип развертывания, предполагающий ручное копирование файлов приложения в целевой каталог на сервере. Он подходит для небольших приложений или когда у вас есть прямой доступ к серверу. Обычно приложение развертывается путем копирования необходимых файлов, включая скомпилированные двоичные файлы, веб-страницы и файлы ресурсов, в указанный каталог. Несмотря на простоту, развертывание файловой системы может быть сопряжено с ошибками и может потребовать дополнительных шагов по настройке.

Пакет веб-развертывания - это сжатый файл, который содержит все файлы и ресурсы, необходимые для развертывания приложения [ASP.NET](https://vk.com/away.php?to=http%3A%2F%2FASP.NET&cc_key=). Он включает в себя предварительно скомпилированные двоичные файлы, параметры конфигурации, файлы содержимого и даже схему базы данных, если это необходимо. Пакеты веб-развертывания создаются с помощью таких инструментов, как Visual Studio, и обеспечивают автоматизированный и последовательный способ развертывания приложений. Пакет может быть развернут на целевом сервере с помощью различных методов, таких как публикация непосредственно из Visual Studio или с помощью инструментов командной строки, таких как MSDeploy.

Web Deploy - это мощная технология развертывания, предоставляемая компанией Microsoft. Она позволяет публиковать [ASP.NET](https://vk.com/away.php?to=http%3A%2F%2FASP.NET&cc_key=) приложение непосредственно из Visual Studio на удаленный сервер. Web Deploy использует комбинацию синхронизации файлов и управления конфигурацией для эффективного развертывания файлов приложения, баз данных и настроек. Он поддерживает инкрементные обновления, что означает, что передаются только изменения с момента последнего развертывания, что сокращает время развертывания. Web Deploy также предоставляет расширенные возможности, такие как развертывание баз данных, шифрование конфиденциальной информации и поддержка параметров развертывания.

Azure App Service предоставляет полностью управляемую платформу для размещения веб-приложений, включая [ASP.NET](https://vk.com/away.php?to=http%3A%2F%2FASP.NET&cc_key=). Имеется возможность опубликовать свое приложение непосредственно из Visual Studio в Azure App Service, а Azure позаботится о базовой инфраструктуре, масштабировании и балансировке нагрузки. Поддерживается непрерывное развертывание из различных систем контроля исходных кодов и предоставляются такие функции, как слоты развертывания для постановки и тестирования, автоматическое масштабирование и мониторинг.

В приложениях [ASP.NET](https://vk.com/away.php?to=http%3A%2F%2FASP.NET&cc_key=)  возможно контейнеризировать приложение с помощью Docker и развернуть его в контейнерном оркестраторе. Контейнеризация позволяет упаковать приложение и его зависимости в легкую изолированную среду, обеспечивая согласованность в различных средах развертывания. Она также позволяет легко масштабировать и развертывать несколько экземпляров приложения.

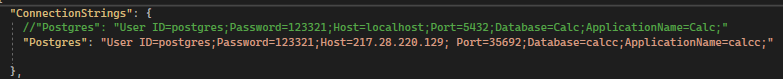
Веб-развертывание IIS: Internet Information Services (IIS) - это популярный веб-сервер для размещения приложений [ASP.NET](https://vk.com/away.php?to=http%3A%2F%2FASP.NET&cc_key=). Вы можете использовать IIS Web Deploy для публикации вашего приложения непосредственно на сервере IIS. Этот метод обеспечивает бесшовную интеграцию с IIS и позволяет настраивать различные параметры развертывания, такие как пулы приложений, методы аутентификации и сертификаты SSL. IIS Web Deploy также поддерживает инкрементные обновления, что делает его эффективным для развертывания обновлений приложений.

Для данной работы был выбрал сервис deploy-f. Данная платформа осуществляет запуск контейнеров Docker или приложений из zip-архива с возможностью масштабирования на лету и управления потреблением ресурсов каждого приложения.

Необходимо собрать и опубликовать приложение с помощью команды "dotnet publish -c Release -o publish" для дальнейшего его развертывания. "dotnet" обозначает инструмент интерфейса командной строки (CLI) .NET, который используется для сборки, запуска и управления приложениями .NET. "publish" - это команда в инструменте dotnet CLI, которая указывает ему опубликовать приложение. "-c Release" указывает конфигурацию сборки, которую следует использовать, в данном случае "Release". Конфигурация сборки определяет, как будет собираться приложение, включая оптимизацию и отладочную информацию. Конфигурация "Release" обычно создает версию, оптимизированную для развертывания, с включенными оптимизациями и исключенными отладочными символами. "-o publish" - указывает выходной каталог для опубликованного приложения. В данном случае каталог называется "publish". Опубликованное приложение будет сгенерировано в этом каталоге. После выполнения данной команды в командной строке из папки с приложением, dotnet CLI построит приложение [ASP.NET](https://vk.com/away.php?utf=1&to=http%3A%2F%2FASP.NET) на основе конфигурации сборки Release и выведет опубликованные файлы в каталог "publish". Опубликованные файлы включают необходимые исполняемые файлы, библиотеки DLL и любые другие файлы, необходимые для запуска приложения в среде развертывания.

После архивации данного каталога "publish" и загрузки его на deploy-f, на стороне сервиса произойдет подготовка сервера и запуск приложения на хостинге.

Также необходимо развернуть базу данных PostgreSQL. Определив текущую версию PostgreSQl, введем имя пользователя, пароль и название создаваемой базы данных. Все автоматически произойдет на стороне сервиса. Будет выдан ip-addres и порт. Для взаимодействия с этой бд необходимо изменить строку подключения, которая находится в appsetting.json. Новая строка подлючения представлена на рисунке 7.1.



***Рисунок 7.1 Строка подключения.***

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.
* Приложение развернуто на удаленном сервере.

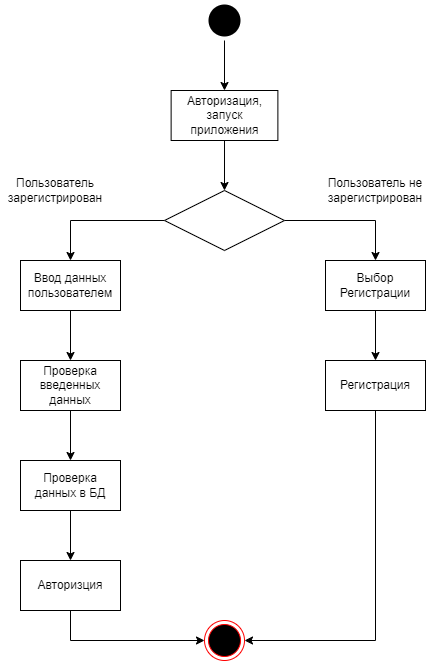
Результатом данной курсовой работы является приложение для расчета калорийности продуктов и блюд. Разработка данного приложения для отслеживания калорий представляет собой ценный вклад в область здоровья и благополучия. Предоставляя пользователям инструменты, необходимые для контроля за потреблением 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

1. Сайт с рейтингом приложений для похудения LifeHacker [Электронный ресурс] URL: https://lifehacker.ru/13-prilozhenij-dlya-poxudeniya/.
2. Руководство по технологии ASP.NET Core Metanit [Электронный ресурс] URL: https://metanit.com/sharp/aspnet6/.
3. Сайт с помощью в создании css-элементов FreeFrontend [Электронный ресурс] URL: https://freefrontend.com/css-buttons/.
4. Сайт про то, что же такое бизнес-процессы SkillBox [Электронный ресурс] URL: https://skillbox.ru/media/management/bolshoy-gayd-ob-upravlenii-biznesprotsessami-glavnoe-chto-dolzhen-znat-kazhdyy-menedzher/.
5. Размещение и развертывание ASP.NET Core [Электронный ресурс] URL: https://learn.microsoft.com/ru-ru/aspnet/core/host-and-deploy/?view=aspnetcore-7.0.\
6. [К. Дж. Дейт](https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D0%B9%D1%82,_%D0%9A%D1%80%D0%B8%D1%81%D1%82%D0%BE%D1%84%D0%B5%D1%80). Introduction to Database Systems. — 8-е изд. — М.: Вильямс, 2006. — С. 1328.
7. ASP.NET Core: Развертывание веб-приложения в службе приложений Azure с помощью Visual Studio [Электронный ресурс] URL: https://habr.com/ru/companies/microsoft/articles/314252/
8. ASP.NET Core Blazor JavaScript interoperability (JS interop) [Электронный ресурс] URL: https://learn.microsoft.com/ru-ru/ASPNET/core/blazor/javascript-interoperability/?view=aspnetcore-8.0
9. Основы JavaScript для начинающих разработчиков [Электронный ресурс] URL: https://habr.com/ru/companies/ruvds/articles/416375/

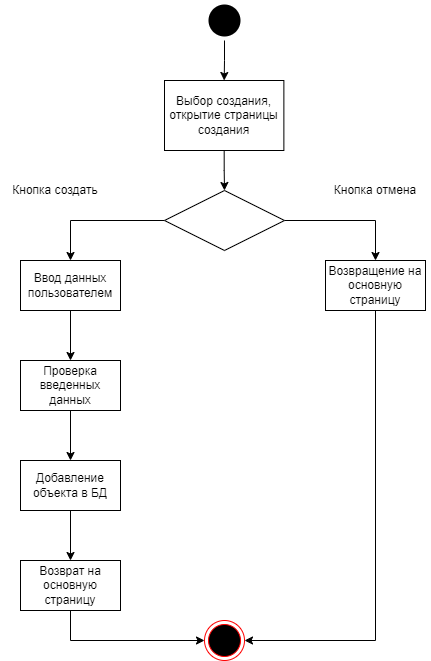
Приложение A. Диаграммы активностей



***Рисунок А.1. Диаграмма активности авторизация***



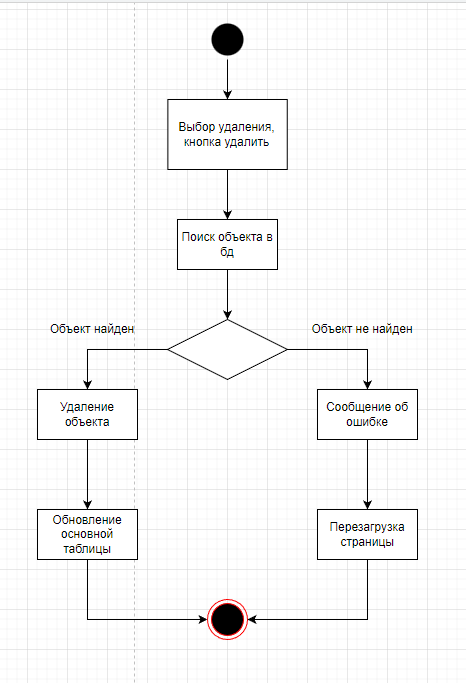
***Рисунок А.2. Диаграмма активности регистрация***



***Рисунок А.3. Диаграмма активности создание***

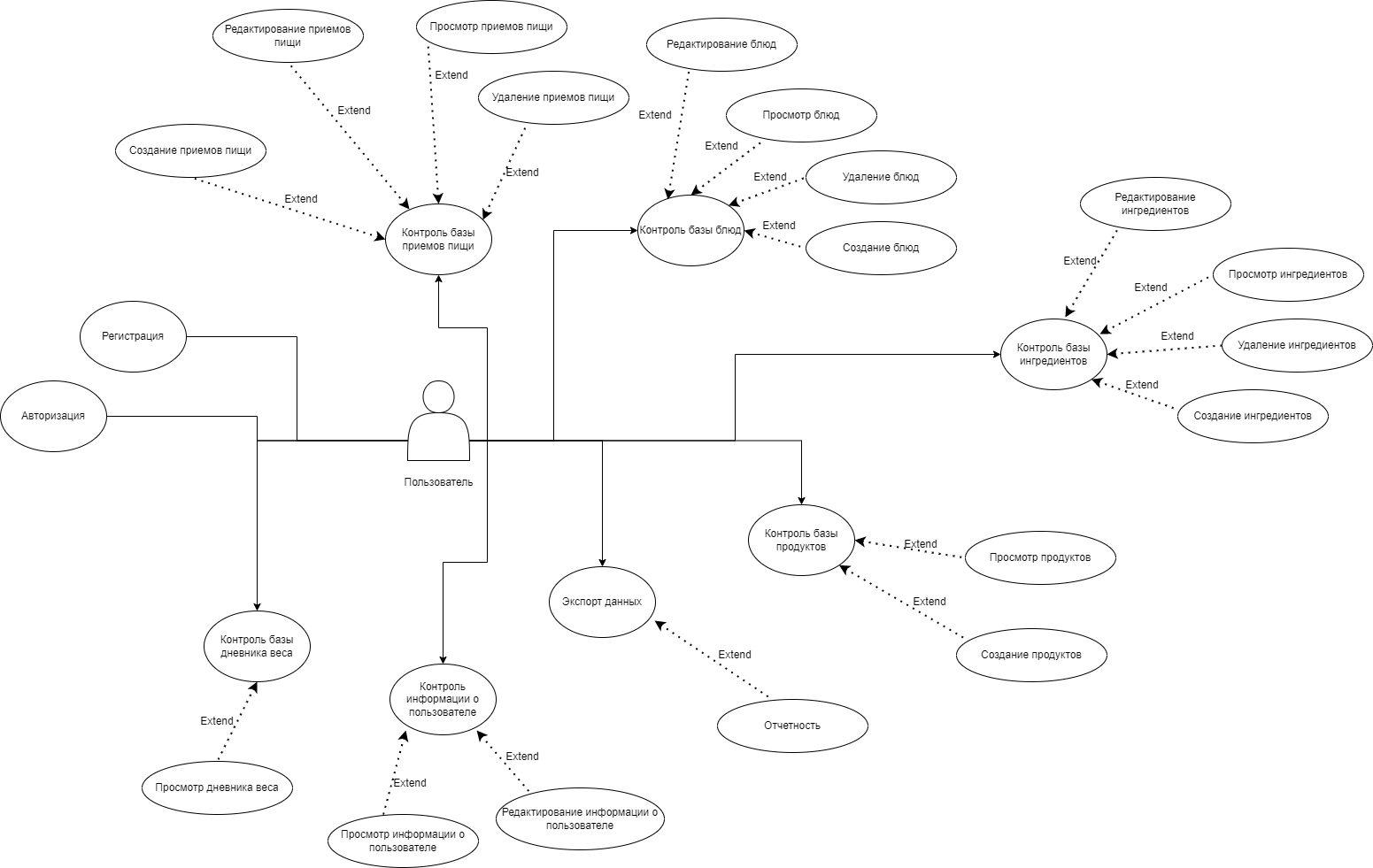


***Рисунок А.4. Диаграмма активности редактирование***



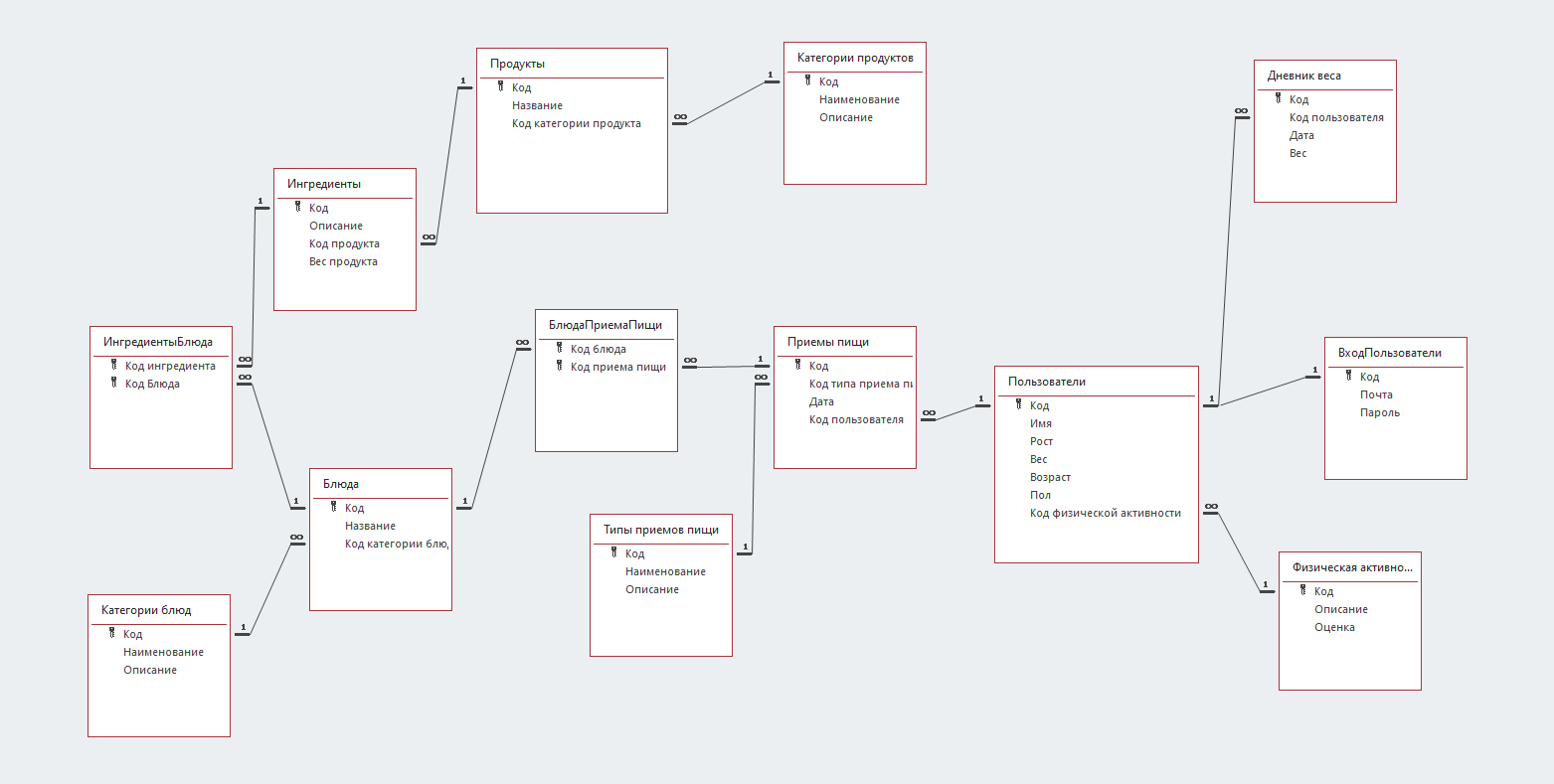
***Рисунок А.5. Диаграмма активности удаление***

Приложение Б. Диаграмма прецедентов



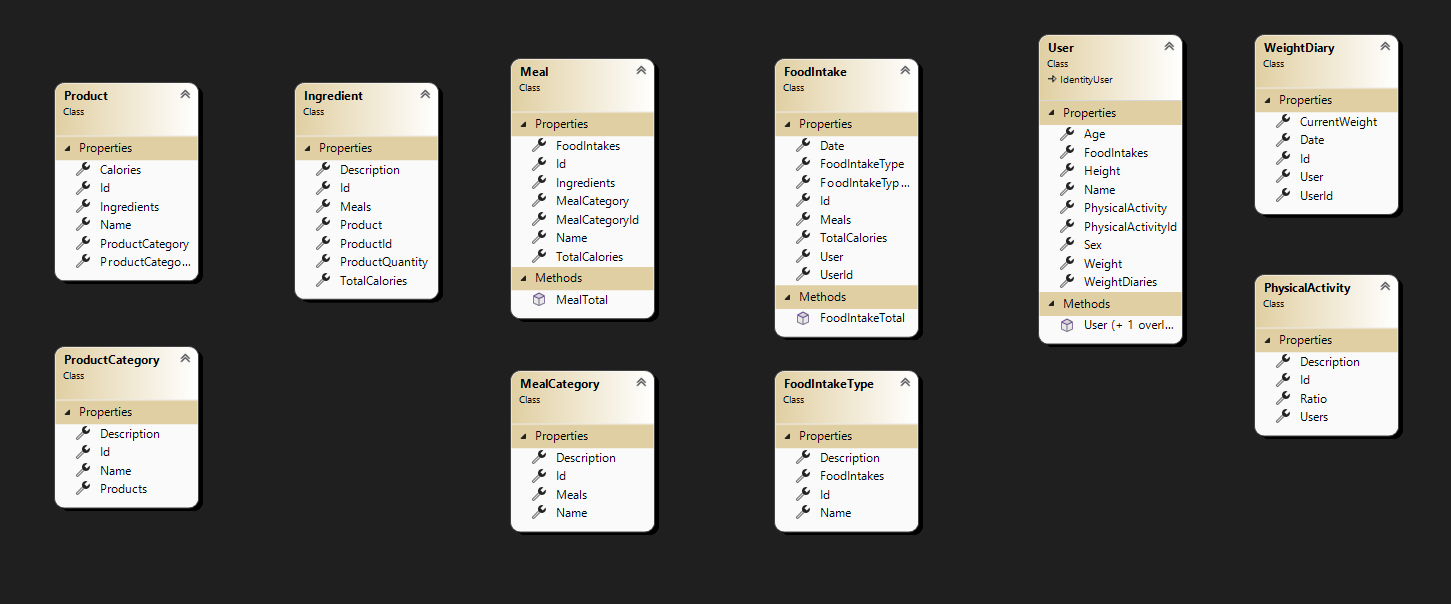
***Рисунок Б.1. Диаграмма прецедентов для приложения для расчета калорийности продуктов и блюд***

Приложение В. ER-диаграмма базы данных



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

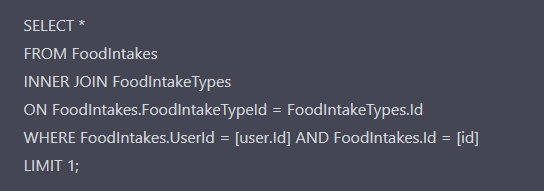
Приложение Г. Диаграмма классов приложения



***Рисунок Г.1. Диаграмма классов приложения для расчета калорийности продуктов и блюд***

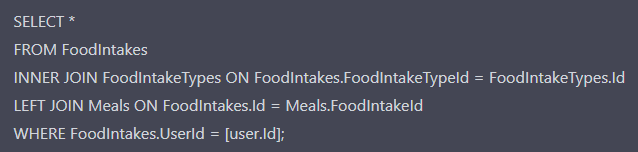
Приложение Д. SQL-запросы

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



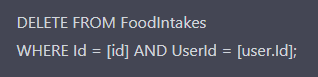
***Рисунок Д.1. Запрос на получение приема пищи***

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



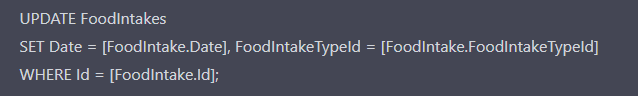
***Рисунок Д.2. Запрос на получение приемов пищи***

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



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

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



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

Приложение Е. Код программы

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

Приложение Ж. Техническое задание

1 Название

Приложение для расчета калорийности продуктов и блюд.

2 Цель проекта

Цель – создать приложение, с помощью которого можно рассчитывать калорийность продуктов и блюд и следить за своим питанием.

3 Информация о заказчике

Заказчик – доцент кафедры информационных технологий в бизнесе Замятина Елена Борисовна.

4 Сроки реализации

Проектные цели необходимо реализовать в течение 10-23.05.2023.

5 Функциональные требования

Приложение должно обладать следующими функциями:

* 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, 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.

6 Требования к исполнителю

Исполнителем является студент второго курса бакалавриата направления подготовки 09.03.04 «Программная инженерия».

7 Этапы и график реализации проекта

Длительность проекта определяется итерационным характером модели жизненного цикла продукта и линейным характером жизненного цикла проекта.

С точки зрения управления проектом выделяются стадии:

* инициация;
* планирование;
* исполнение;
* завершение.

С точки зрения управления жизненным циклом продукта проект состоит из трех итераций, каждая из которых состоит из пяти этапов:

* инициализация.
* проектирование.
* реализация.
* тестирование.
* развертывание.

8 Критерии завершенности проекта

Критерием завершенности проекта является наличие работоспособного программного обеспечения, принятого комиссией.

9 Критерии качества продукта

Критерии качества продукта отражены в таблице Ж.1.

***Таблица Ж.1 – Критерии качества продукта***

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Критерий** | **Подтверждение** | **Шкала** |
| 1 | Наличие работоспособной программы «Приложение для расчета калорийности продуктов и блюд» | Сценарии тестирования | Да/нет |
| 3 | Наличие проектной документации | Проектная заявка | 0-3 |
| Техническое задание |
| Руководство пользователя |
| 4 | Регулярное взаимодействие с Заказчиком | Оценочный лист | Да/нет |
| 5 | Отчет о завершении Проекта | Презентация | Да/нет |

10 Форма отчетности

К защите проекта предоставить следующий комплект программной документации:

* техническое задание;
* руководство пользователя;
* персональный отчет по проектной работе;
* исходный код приложения.

11 Трудоемкость проекта

Итоговая трудоемкость проекта составляет 5 з.е.

Приложение И. Руководство пользователя

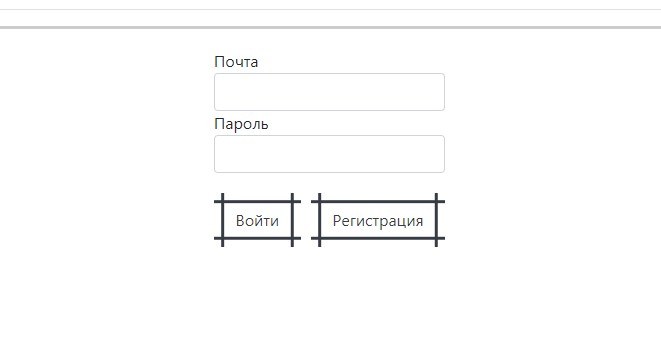
**Назначение системы для расчета калорийности продуктов и блюд**

Приложение для расчета калорийности продуктов и блюд предназначено для людей, которые хотят следить за своим рационом и контролировать количество потребляемых калорий. Данное приложение помогает пользователю оценить калорийность продуктов и блюд, что позволяет им принимать более осознанные решения в отношении своего питания. Назначение системы заключается в том, чтобы облегчить процесс отслеживания пищевого рациона и помочь людям достигнуть своих целей в отношении веса и здоровья.

**Инструкция для взаимодействия с приложением**

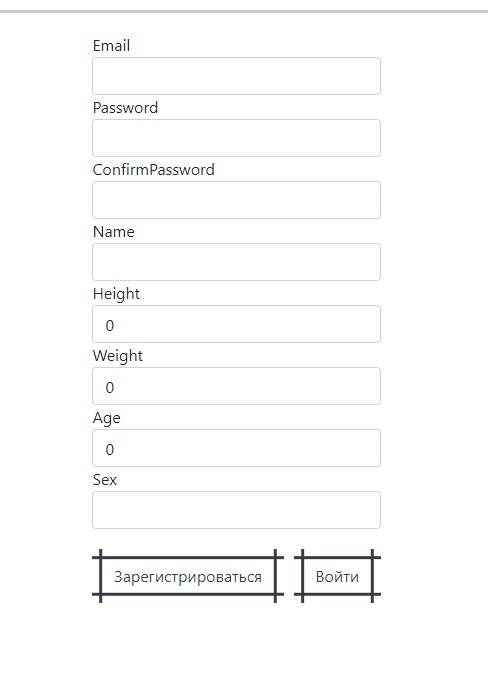
Для начала взаимодействия с приложением вам будет необходимо перейти по URL адресу приложения.

Вам будем предложено ввести данные, необходимые для авторизации, такие как почта и пароль. Страница авторизации представлена на рисунке И.1.



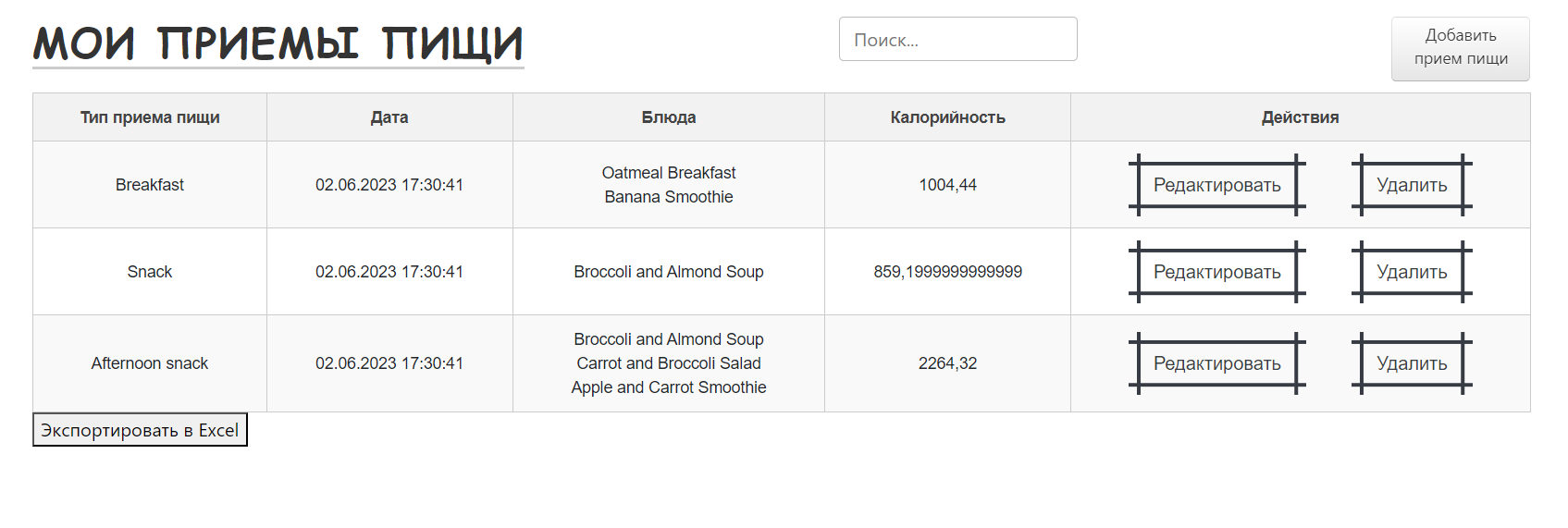
***Рисунок И.1. Страница авторизации***

Если Вы уже авторизированы на данном сайте, то после корректного ввода данных вам откроется доступ к приложению. Если же Вы впервые пользуетесь данной системой, то необходимо нажать на кнопку Регистрации и перейти на страницу регистрации, которая представлена на рисунке И.2. Корректно пройдя регистрацию, Вам откроется доступ к приложению.



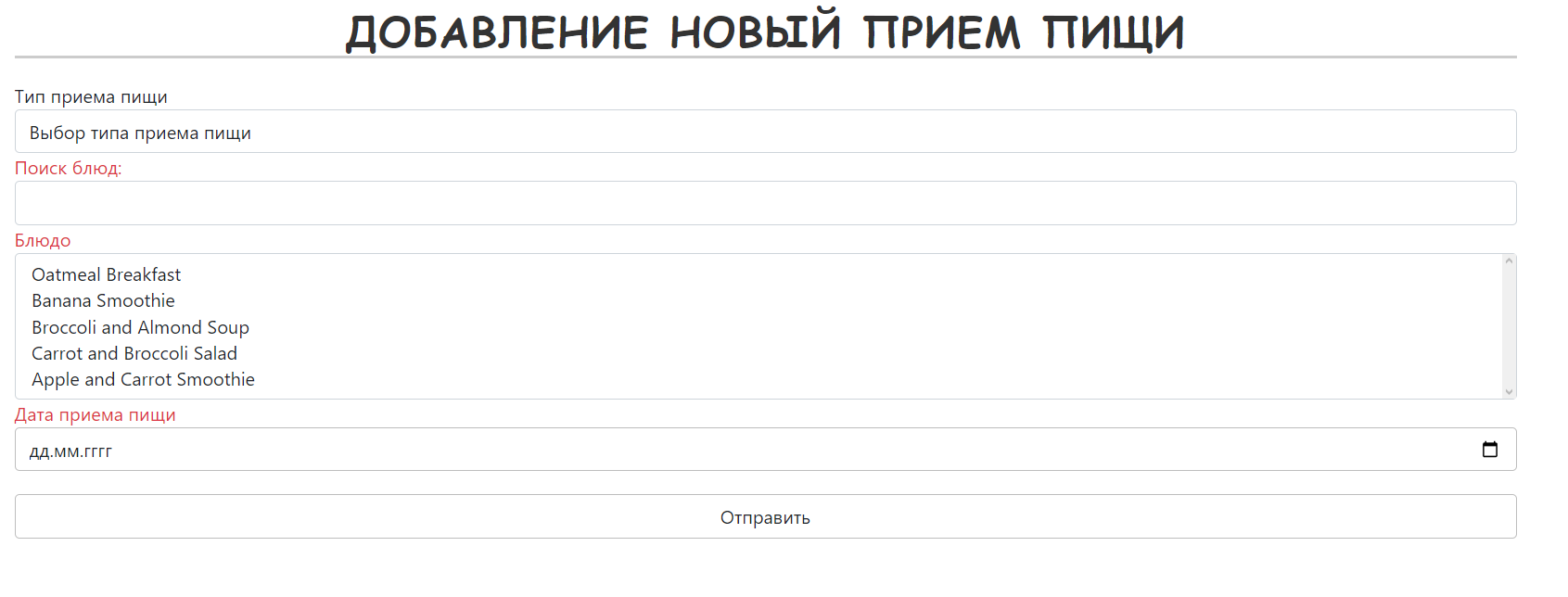
***Рисунок И.2. Страница регистрации***

После успешной авторизации/регистрации Вас встретит главная страница приложения, на которой изображены Ваши приемы пищи и информация о них. Главная страница приложения представлена на рисунке И.3.



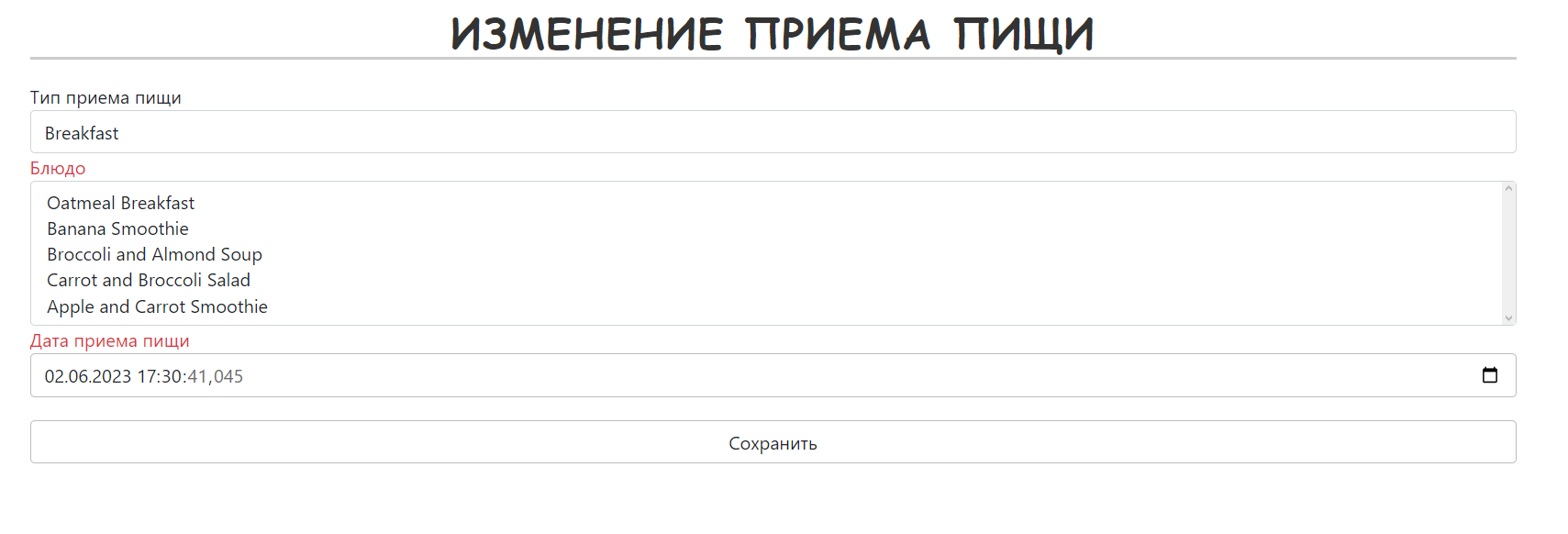
***Рисунок И.3. Страница приемов пищи***

Если Вы хотите добавить новый прием пищи, Вам необходимо нажать на кнопку Добавить прием пищи, после чего откроется страница Создания Приема пищи, на которой Вы введете необходимые данные. Для создания необходимо нажать кнопку Отправить, для отмены кнопку CalcCalories в хедере, после нажатия кнопку Вам откроется страница приемов пищи. Страница добавления приема пищи представлена на рисунке И.4.



***Рисунок И.4. Страница добавления приемов пищи***

Если Вы хотите изменить существующий прием пищи, нажмите на Редактировать. Приложение переадресует Вас на страницу редактирования. Вы вводите данные и нажимаете сохранить. Страница редактирования представлена на рисунке И.5.



***Рисунок И.5. Страница редактирования приема пищи***

Если Вы хотите удалить прием пищи, нажмите на кнопку Удалить у того приема пищи, который желаете удалить.