ззз

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждения образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных Технологий

Кафедра Программной инженерии

Специальность 1-40 05 01-03 Информационные системы и технологии

Направление специальности 1-40 05 01-03 Информационные системы и технологии

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**КУРСОВОГО ПРОЕКТА:**

по дисциплине «Объектно-ориентированное программирование»

Тема Программное средство «Индивидуальный тренер»

Исполнитель

студентка 2 курса группы 3 Кохнюк Александра Сергеевна

(Ф.И.О.)

Руководитель работы Сухорукова И. Г.

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

Курсовой проект защищен с оценкой

Председатель Пацей Н.В.

(подпись)

Минск 2022

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПЕУБЛИКИ БЕЛАРУСЬ

Учреждение образования   
«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий   
Кафедра программной инженерии

Утверждаю

Заведующий кафедрой ПИ

\_\_\_\_\_\_\_\_\_\_\_\_ \_\_ Н.В. Пацей

подпись инициалы и фамилия

“\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2022г.

**ЗАДАНИЕ**

**к курсовому проектированию**

**по дисциплине** "Объектно-ориентированное программирование"

|  |  |
| --- | --- |
| Специальность: 1-40 05 01-03 Информационные системы и технологии (издательско-полиграфический комплекс)  Студент: Кохнюк А. С. | Группа:\_\_3\_\_ |
| **Тема: Программное средство «Индивидуальный тренер»** | |

**1. Срок сдачи студентом законченной работы**: "20 мая 2022 г."

**2. Исходные данные к проекту:**

**2.1**. Функционально ПС поддерживает:

* Регистрация и авторизация пользователей;
* Поиск и фильтрация тренировок;
* Выбор тренировки из меню;
* Добавление и удаление тренировок;
* Добавление в избранное;
* Составление статистики работы за неделю;

**2.2.** При выполнении курсового проекта необходимо использовать принципы проектирования ООП. Приложение разрабатывается под ОС Windows и представляет собой настольное приложение. Отображение, бизнес логика должны быть максимально независимы друг от друга для возможности расширения. Диаграммы вариантов использования, классов реализации задачи, взаимодействия разработать на основе UML. Язык разработки проекта – C#. Управление программой должно быть интуитивно понятным и удобным. При разработке использовать несколько наиболее подходящих шаблонов проектирования ПО.

**3. Содержание расчетно-пояснительной записки**

(перечень вопросов подлежащих разработке)

* Введение
* Постановка задачи и обзор литературы (алгоритмы решения, обзор прототипов, актуальность задачи)
* Проектирование архитектуры проекта (структура модулей, классов).
* Разработка функциональной модели и модели данных программного средства (выполняемые функции)
* Тестирование
* Заключение
* Список используемых источников
* Приложения

**4. Форма представления выполненной курсовой работы:**

* + Теоретическая часть курсового проекта должны быть представлены в формате docx. Оформление записки должно быть согласно выданным правилам.
  + Листинги программы представляются частично в приложении.
  + Пояснительную записку, листинги, проект (инсталляцию проекта) необходимо загрузить диск, указанный преподавателем.

#### Календарный план

|  |  |  |  |
| --- | --- | --- | --- |
| №  п/п | Наименование этапов курсового проекта | Срок выполнения этапов проекта | Примечание |
| 1 | Введение | 19.02.2022 |  |
| 2 | Аналитический обзор литературы по теме проекта. Изучение требований, определение вариантов использования | 12.03.2022 |  |
| 3 | Анализ и проектирование архитектуры приложения (построение диаграмм, проектирование бизнес-слоя, представления и данных) | 26.03.2022 |  |
| 4 | Проектирование структуры базы данных. Разработка дизайна пользовательского интерфейса | 2.04.2022 |  |
| 5 | Кодирование программного средства | 23.04.2022 |  |
| 6 | Тестирования и отладка программного средства | 30.04.2022 |  |
| 7 | Оформление пояснительной записки | 7.05.2022 |  |
| 9 | Сдача проекта | 20.05.2022 |  |

**5. Дата выдачи задания \_\_\_\_**14.02.2022**\_\_\_\_**

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *И. Г. Сухорукова*

(подпись)

Задание принял к исполнению *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. С*. *Кохнюк*

(дата и подпись студента)

#### ВВЕДЕНИЕ

Приложение «Индивидуальный тренер» является программным средством для того, чтобы дать возможность пользователям в удобной форме заниматься спортом в любое время и удобном месте.

В современном мире многие люди пытаются вести спортивный образ жизни, однако не каждый человек может позволить себе посещать тренажерные залы. Для этого и было создано данное ПО.

В качестве интерфейса приклaдного программирования был выбран обширный API-интерфейс — Windows Presentation Foundation (WPF), предназначенный для создания настольных программ с графически насыщенным пользовательским интерфейсом.

Для работы с WPF использовался объектно-ориентированный язык программирования с С-подобным синтаксисом — С#, разработанный для создания приложений на платформе Microsoft .NET Framework.

Хранение данных осуществляется в Microsoft SQL Server.

## **Аналитический обзор литературы**

**«Фитнес Бро»** - приложение, помогающее людям заниматься спортом. Каталог упражнений позволяет ознакомиться и выбрать наилучшие упражнения для тренировки. Любимые упражнения могут быть сохранены в отдельном месте, чтобы всегда иметь к ним быстрый доступ. Секундомер поможет засекать время между подходами и не даст тебе "остыть". Добавлен раздел напоминаний - теперь сложно забыть выпить витамин. Ниже на рисунке *1.1* представлен внешний вид приложения.

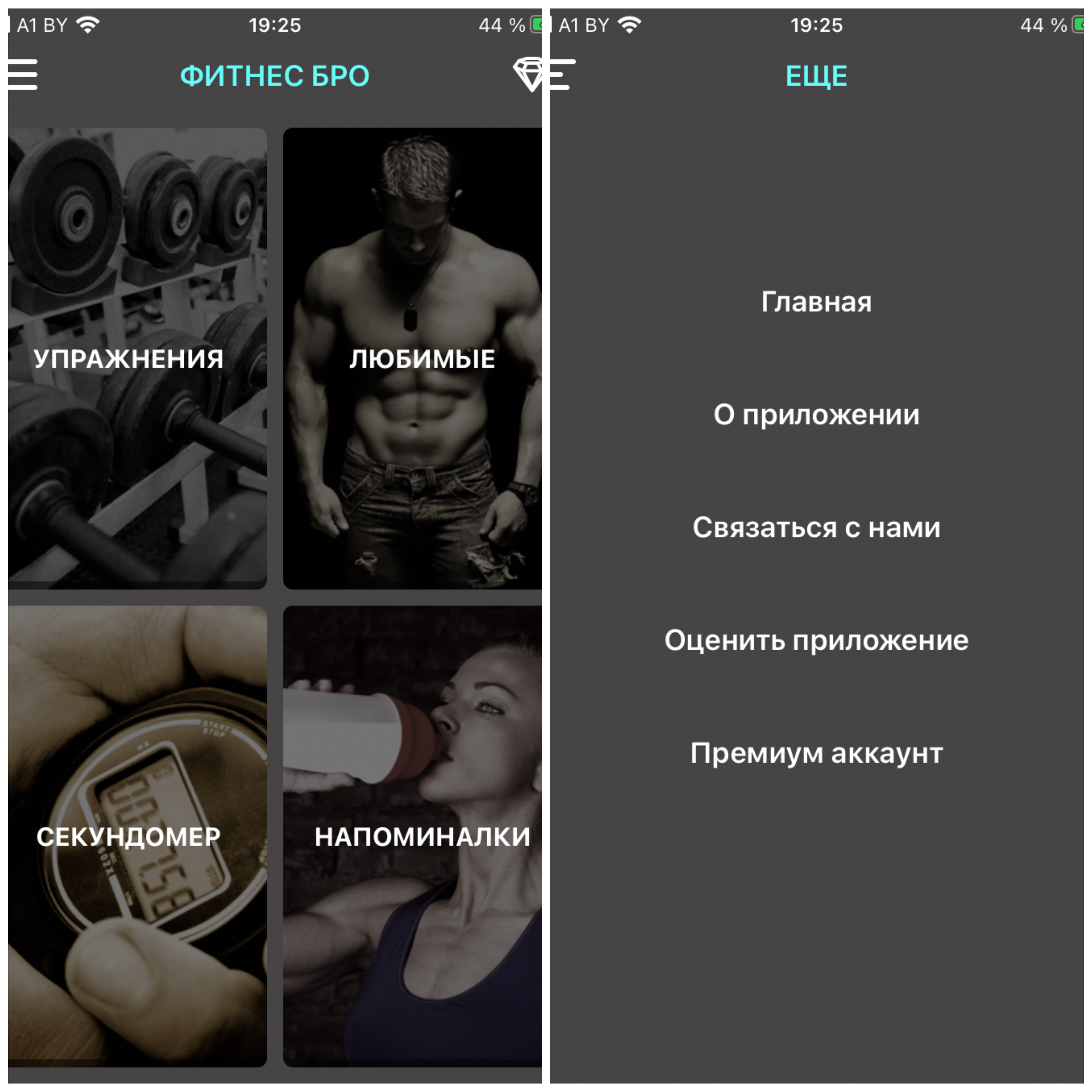


Рис. 1.1 – Приложения «Фитнес Бро»

Приложение **«Fitabillity»** - авторская программа персональных тренировок, разработанная известным в спортивных кругах тренером по функциональной подготовке, реабилитологом Марией Буровой. Это уникальная сбалансированная система самостоятельных тренировок, основанная на десятилетнем профессиональном опыте действующего тренера и международных медицинских разработках. Приложение дает возможность выбора тренировки на какую-либо отдельную часть тела, где можно скачать видео с инструкциями тренировок. Ниже на рисунке *1.2* представлен внешний вид приложения.

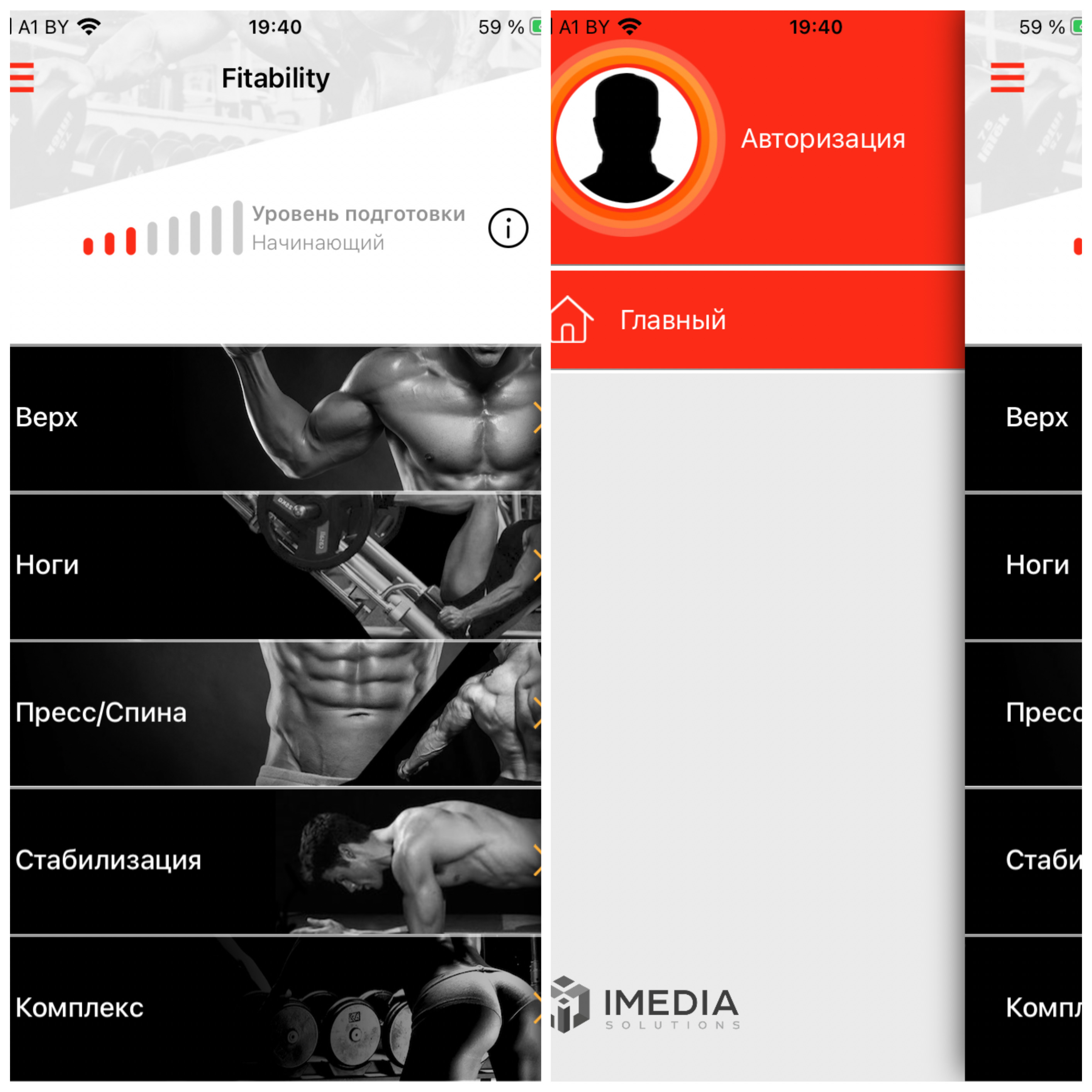


Рис. 1.2 – Приложения «Fitabillity»

**«Freeletics»** - фитнес-приложение, дающее возможность заниматься с лучшим цифровым тренером где угодно. ПО представляет из себя систему, состоящую из 22 тренировки с собственным весом, 30 видов упражнений с изображениями, а также 20 аудиосессий. Ниже на рисунке *1.3* представлен внешний вид приложения.

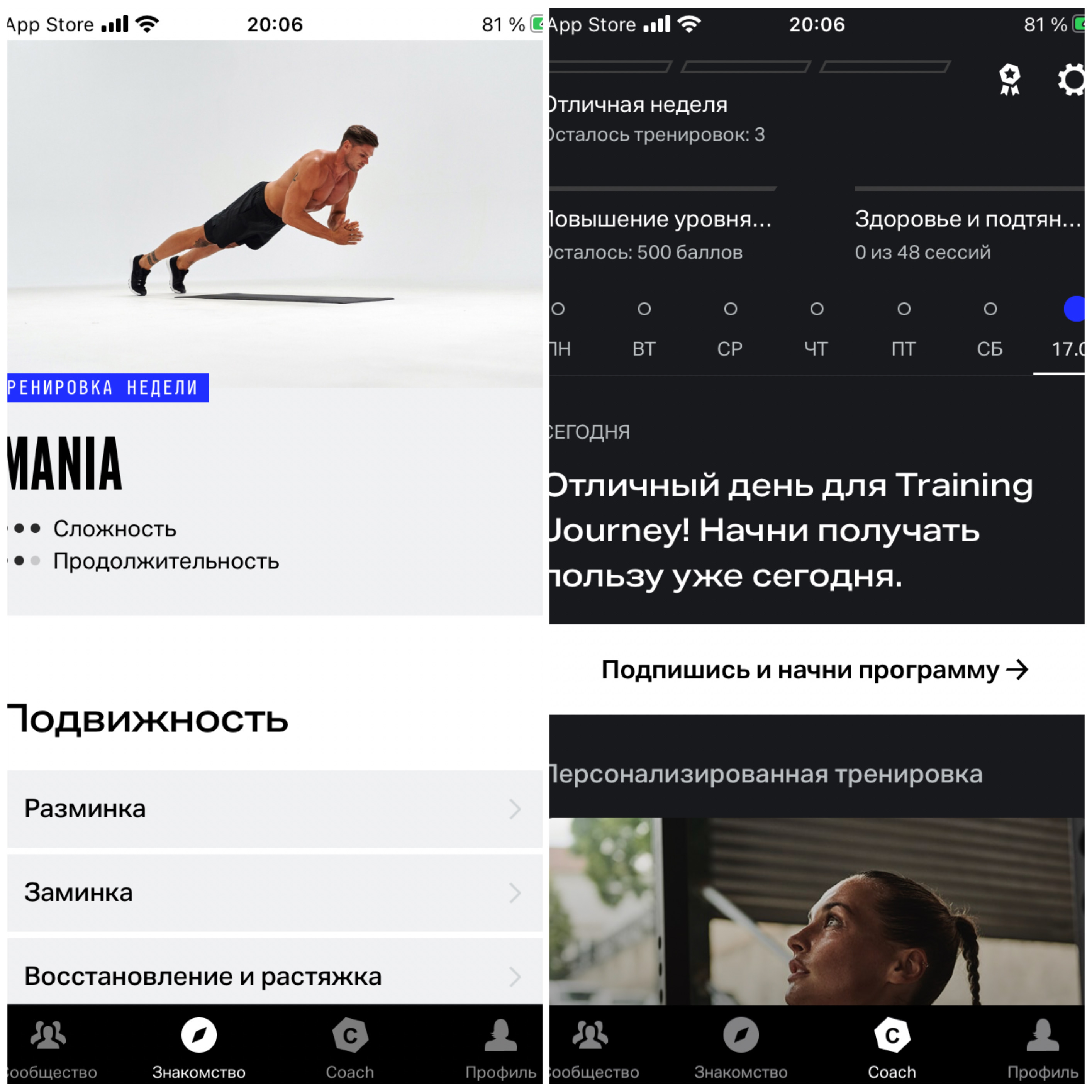


Рис. 1.3 – Приложения «Freeletics»

#### Анализ требований к программному средству и разработка функциональных требований

Анализ требований — это процесс сбора требований к программному обеспечению, их систематизации, документирования, анализа, выявления противоречий, неполноты, разрешения конфликтов в процессе разработки программного обеспечения.

Цель анализа требований в проектах — получить максимум информации, уточнить рамки проекта, оценить возможные риски. На этом этапе происходит идентификация принципиальных требований методологического и технологического характера, формулируются цели и задачи проекта, а также определяются критические факторы успеха, которые впоследствии будут использоваться для оценки результатов внедрения. Определение и описание требований — шаги, которые во многом определяют успех всего проекта, поскольку именно они влияют на все остальные этапы.

Задачей курсового проекта является разработка приложения, предназначенного для занятия спортом, а также ряда дополнительных операций: добавления юзеров, возможность администратора добавления и удаления диет и тренировок; добавление в избранное диет и тренировок пользователем; вывод информации и о тренировках и диетах.

#### Описание средств разработки

Программный продукт должен быть реализовать на объектно-ориентированном языке программирования C# с использованием технологии WPF. Windows Presentation Foundation (WPF) — система для построения клиентских приложений Windows с визуально привлекательными возможностями взаимодействия с пользователем, графическая (презентационная) подсистема в составе .NET Framework (начиная с версии 3.0), использующая язык XAML.

В основе WPF лежит векторная система визуализации, не зависящая от разрешения устройства вывода и созданная с учётом возможностей современного графического оборудования. WPF предоставляет средства для создания визуального интерфейса, включая язык XAML (eXtensible Application Markup Language), элементы управления, привязку данных, макеты, двухмерную и трёхмерную графику, анимацию, стили, шаблоны, документы, текст, мультимедиа и оформление.

База данных была разработана в Microsoft SQL Server.

SQL Server является одной из наиболее популярных систем управления базами данных (СУБД) в мире.

SQL Server характеризуется такими особенностями как:

* Производительность. SQL Server работает очень быстро.
* Надежность и безопасность. SQL Server предоставляет шифрование данных.
* Простота. С данной СУБД относительно легко работать и вести администрирование.

Для организации баз данных MS SQL Server использует реляционную модель, которая предполагает хранение данных в виде таблиц, каждая из которых состоит из строк и столбцов. Каждая строка хранит отдельный объект, а в столбцах размещаются атрибуты этого объекта.

Для взаимодействия с базой данных применяется язык SQL (Structured Query Language). Клиент (например, внешняя программа) отправляет запрос на языке SQL должным образом интерпретирует и выполняет запрос, а затем посылает клиенту результат выполнения.

Основной используемый язык запросов —Transact-SQL — реализован на структурированном языке запросов (SQL) с расширениями.

Для осуществления связи между базой данных и приложением на C# необходим посредник, которым будет являться технология ADO.NET. ADO.NET позволяют создавать подключения, команды и прочие объекты для взаимодействия с базами данных.

#### 2. Описание разрабатываемой функциональности приложения

Программное средство предоставляет пользователю следующие функциональные возможности:

* Регистрация и авторизация
* Возможность добавить в избранное тренировку
* Возможность добавить в избранное диету
* Возможность удалить из избранного тренировку
* Возможность удалить из избранного диету
* Поиск тренировки по названию
* Поиск диеты по названию
* Отметить тренировку пройденной
* Отменить отметку о прохождении
* Сформировать отчет о проделанной работе

Администратор может:

* Добавление тренировки
* Удаление тренировки
* Добавление диеты
* Удаление диеты
* Отметить тренировку пройденной
* Отменить отметку о прохождении
* Поиск тренировки по названию
* Поиск диеты по названию
* Сформировать отчет о проделанной работе

В графической части записки представлена UML-диаграмма возможностей, которая отражает функциональность программного средства с точки зрения получения значимого результата для пользователя.

#### Проектирование программного средства

Проектирование программного средства — процесс создания проекта программного обеспечения. Целью проектирования является определение внутренних свойств системы и детализации её внешних свойств на основе исходных условий задачи. Исходные условия задачи уже были сформулированы во втором разделе данной пояснительной записки. Этап проектирования подразумевает их анализ.

#### 3. 1 Обобщенная структура проекта и детализация функций проектируемого программного средства

При запуске приложения открывается главное окно. В нем можно выбрать авторизацию или регистрацию, принцип авторизации и регистрации показан ниже на рисунке *3.1* в блок-схеме.

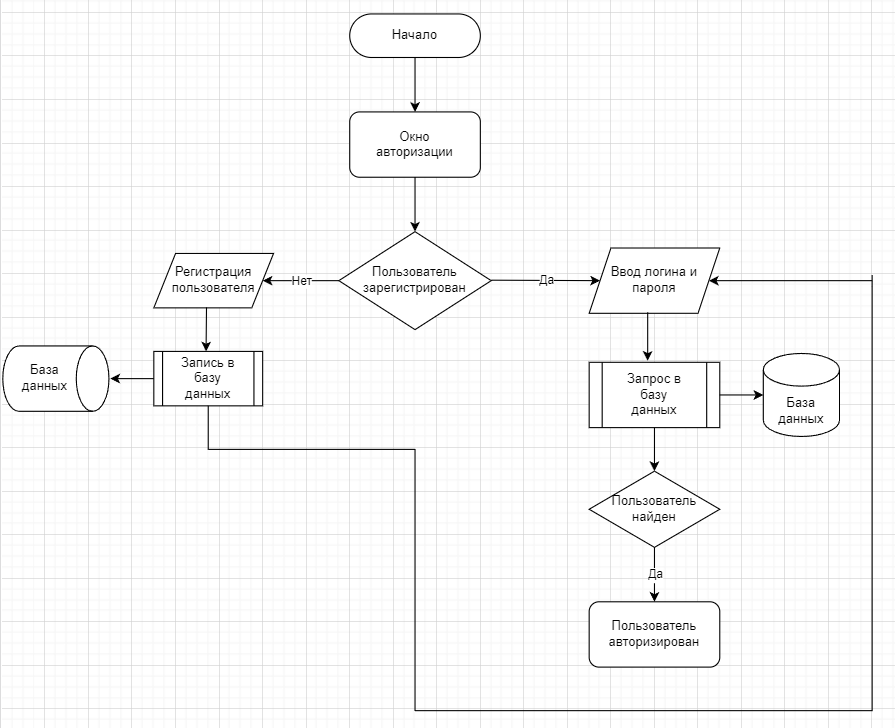


Рис. 3.1 – Блок-схема работы авторизации и регистрации

Структура проекта представлена на рисунке *3.1*

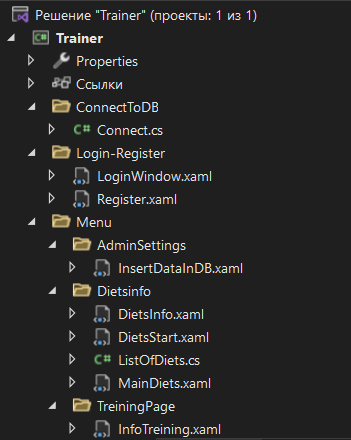
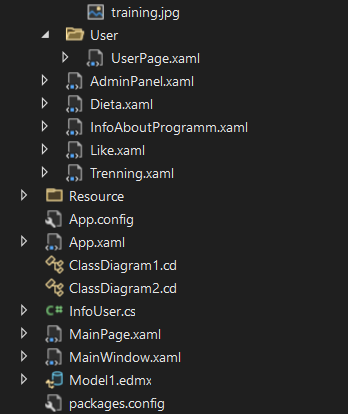
 

Рис. *3.1* – Структура проекта

Описание приведенной структуры проекта представлено в таблице 3.1

*Таблица 3.1*

|  |  |
| --- | --- |
| Имя файла | Содержание |
| ConnectToDB | Репозитории, который непосредственно взаимодействующими с данными, и остальной программой. |
| Properties | Свойства проекта, содержит информацию о сборке используемых ресурсах и настройках. |
| Login-Register | Содержит страницы, которые позволяют пользователю работать с приложением. Они описывают графическую составляющую приложения. |
| Menu | Содержит страницы, которые позволяют пользователю работать с приложением. Они описывают графическую составляющую приложения. |
| User | Содержит страницe, которые позволяют пользователю работать с приложением. Они описывают графическую составляющую приложения |
| Resource | Файлы иконок |
| App.config | Файл конфигурации приложения |
| App.xaml | Основная задача данного файла состоит в определении ресурсов, общих для приложения |

Таким образом, сформированная таблица помогает понять общую стркутура проекта проектируемого программного средства.

Структуры данных, а также их зависимости друг от друга представлены ниже на рисунке *3.2*

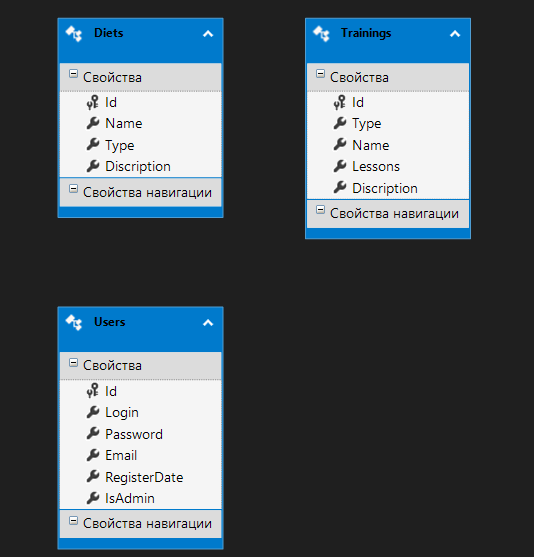


Рис. *3.2* – Структура данных

На рисунке *3.3* представлен проект таблицы «Диеты», которая содержит такие поля как: название диеты, картинка, тип диеты и ее описание.

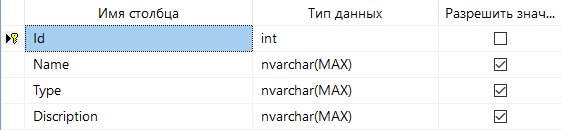


Рис. *3.3* – Структура данных таблицы «Диеты»

На рисунке *3.4* представлен проект таблицы «Избранное», которая содержит такие поля как: идентификационный номер пользователя и тип избранной функции (тренировка или диета).

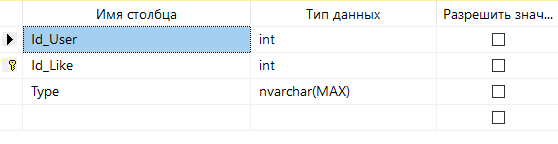


Рис. *3.4* – Структура данных таблицы «Избранное»

На рисунке *3.5* представлен проект таблицы «Тренировки», которая содержит такие поля как: тип тренировки, ее название, количество уроков и описание.

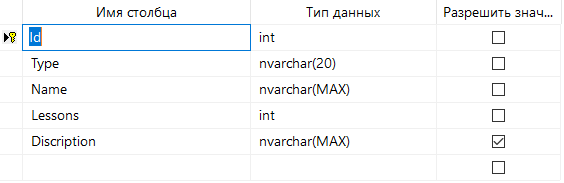


Рис. *3.5* – Структура данных таблицы «Тренировки»

На рисунке *3.6* представлен проект таблицы «Пользователи», которая содержит такие поля как: логин, пароль, адрес электронной почты, дата регистрации и метка, которая показывает полномочия пользователя (1-администратор, 0-обычный пользователь).

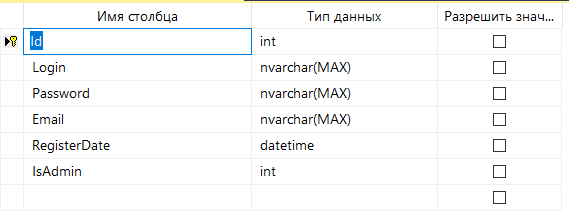


Рис. *3.6* – Структура данных таблицы «Пользователи»

На рисунке *3.* представлен проект таблицы «Пройденные тренировки», которая содержит такие поля как: id пользователя, id тренировки и дату отметки.

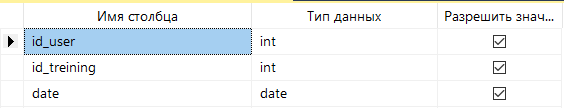


Рис. *3.7* – Структура данных таблицы «Пройденные тренировки»

## **3.2 Пример варианта использования**

Диаграмма классов представлена в **приложении А**.

* 1. **Схема работы приложения**

При запуске приложения открывается главное окно. В нем можно выбрать авторизацию или регистрацию. Если войти в приложение под именем администратора, будет видна панель администратора, в которой он может добавлять и удалять тренировки, добавлять и удалять диеты. Ниже на рисунке *3.7* представлена блок-схема описывающие его возможности.

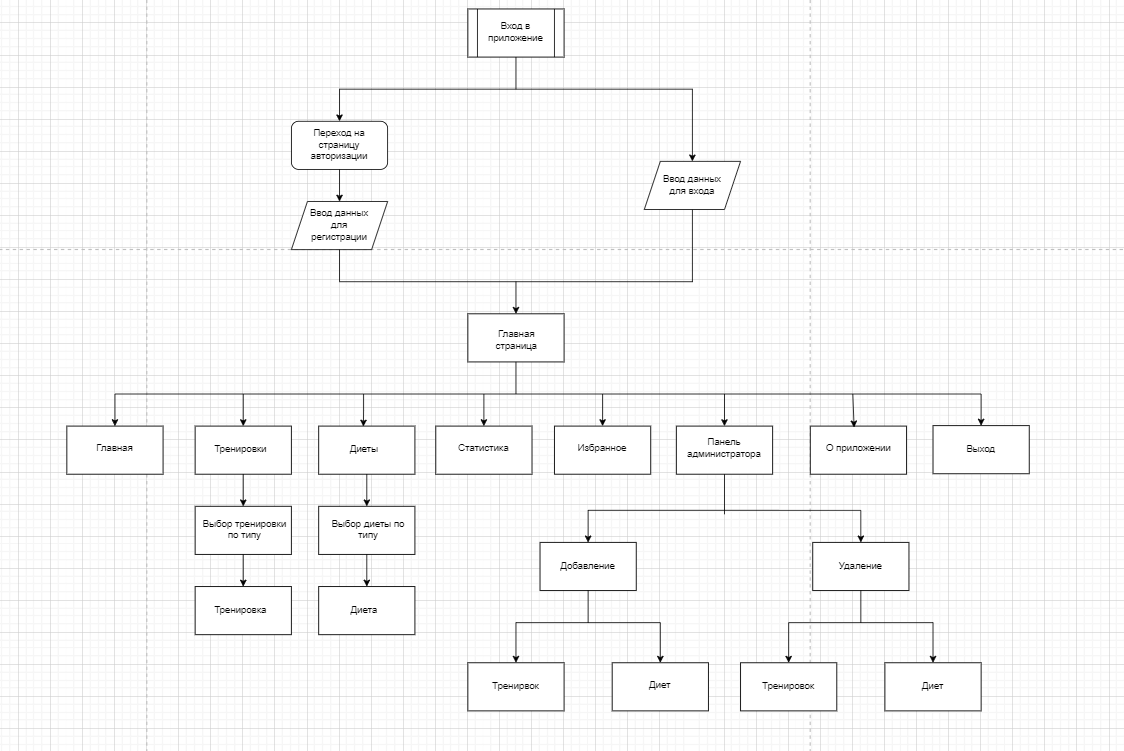


Рис. 3.7 – Блок-схема «Авторизация и возможности администратора»

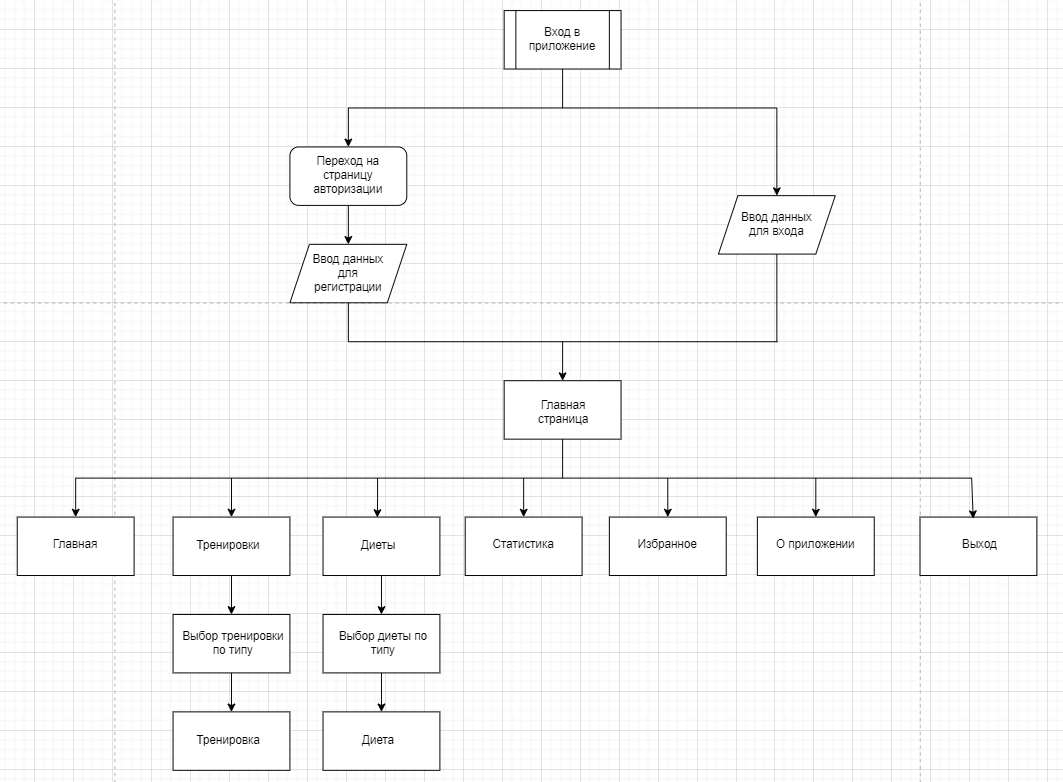


Рис. 3.8 – Блок-схема «Авторизация и возможности пользователя»

На рисунке *3.8* представлена общая схема работы приложения клиента. При запуске приложения у нас появляется окно авторизации, которое требует ввода логина и пароля для начала работы.

В случае, если вас нету своей учётной записи, её можно создать, нажав на кнопку «Регистрация» в окне авторизации. Вы будете направлены в окно регистрации, где вам нужно будет заполнить все поля. После успешной регистрации, вам нужно будет вернутся в окно авторизации и ввести ваши данные, указанные при регистрации.

Нажав на соответствующие кнопки в окне авторизации, у вас появится окно, где нужно будет ввести данные для входа в учётную запись, через которую производится вход.

После ввода логина и пароля в соответствующих полях, проверяется правильность ввода данных.

В случае неверного ввода данных, выведется соответствующее сообщение.

В главном окне работа идёт в страничном и оконных режимах.

## **3.4 Диаграмма последовательностей**

С помощью диаграммы, представленной на рисунке *3.9*, показана последовательность действий при авторизации пользователя.

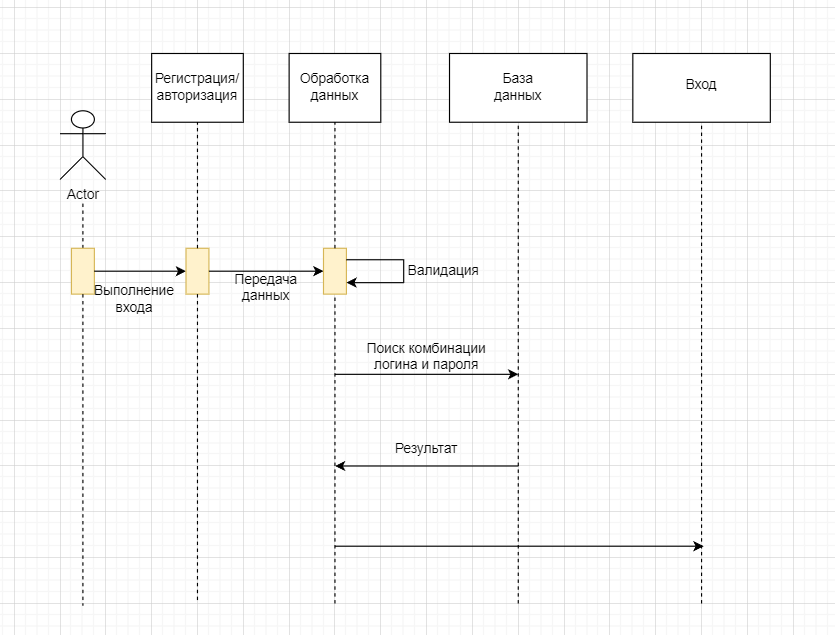


Рис. 3.9 – Диаграмма последовательности

## **Создание (реализация) программного средства**

Следующим этапом разработки приложения является непосредственная реализация программного решения в соответствии с уже сформированными требованиями и шаблонами.

## **Структура проекта и выполняемые функции**

Структура проекта была представлена выше на рисунке *3.1.*

#### Добавление тренировок и диет

Добавление осуществляется с помощью пользовательского элемента управления InsertDataInDB и обработчиков событий AddToDBDiets\_Click(..) и AddToDBTreining\_Click(..). Это представлено на рисунке 4. 1. Полный код представлен в **приложении Б**.

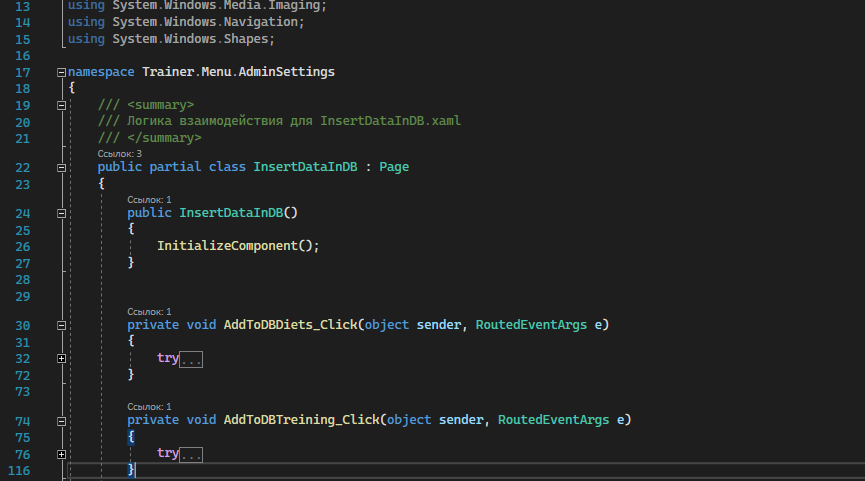


Рис. 4.1 – Структура класса InsertDataInDB

#### Удаление тренировок и диет

Удаление осуществляется с помощью пользовательского элемента управления AdminPanel и обработчика событий Delete\_Click. Это представлено на рисунке 4. 2. Полный код представлен в **приложении В.**



Рис. 4.2 – Структура класса AdminPanel

#### Тестирование, проверка работоспособности и анализ полученных результатов

#### Данное программное средство тестировалось вручную, были выполнены все технические моменты, предусмотренные реализацией, а также были предприняты попытки нарушить работу приложения.

#### Для всех полей ввода текстовой информации предусмотрена валидация данных. Она реализована с помощью классов LoginBox\_TextChanged и passwordBox\_PasswordChanged. При вводе на русском языке данные будут гореть красным цветом и о неккоректности данных сообщит система, как это представлено на рисунке 5.1

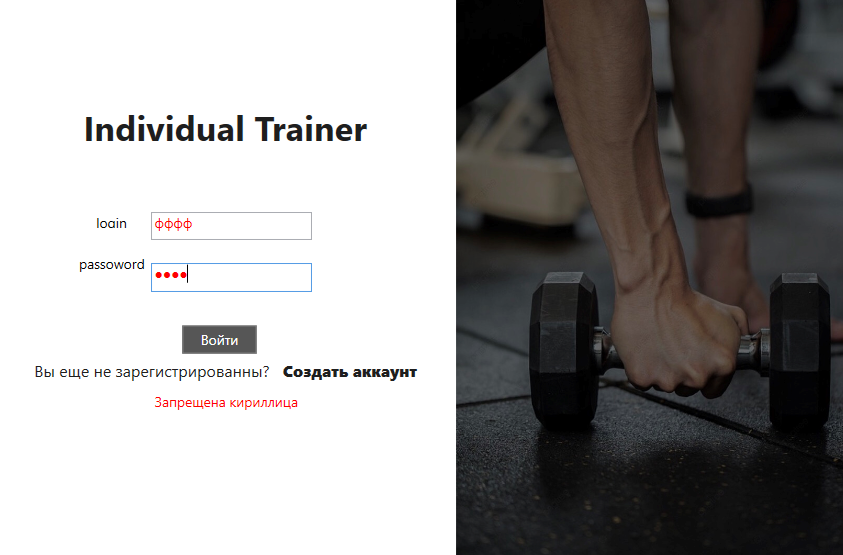


Рис. 5.1 – Окно авторизации

Код для валидации вводимых данных приведен **в приложении Г**

Далее представлено тестирование функций приложения. Если при добавлении тренировки через панель администратора, ввести в поле «Количество уроков» тип которого является числовой, какие-либо символы, система также уведомит о неккоректности вводимых данных. Это представлено на рисунках *5.2* и *5.3*

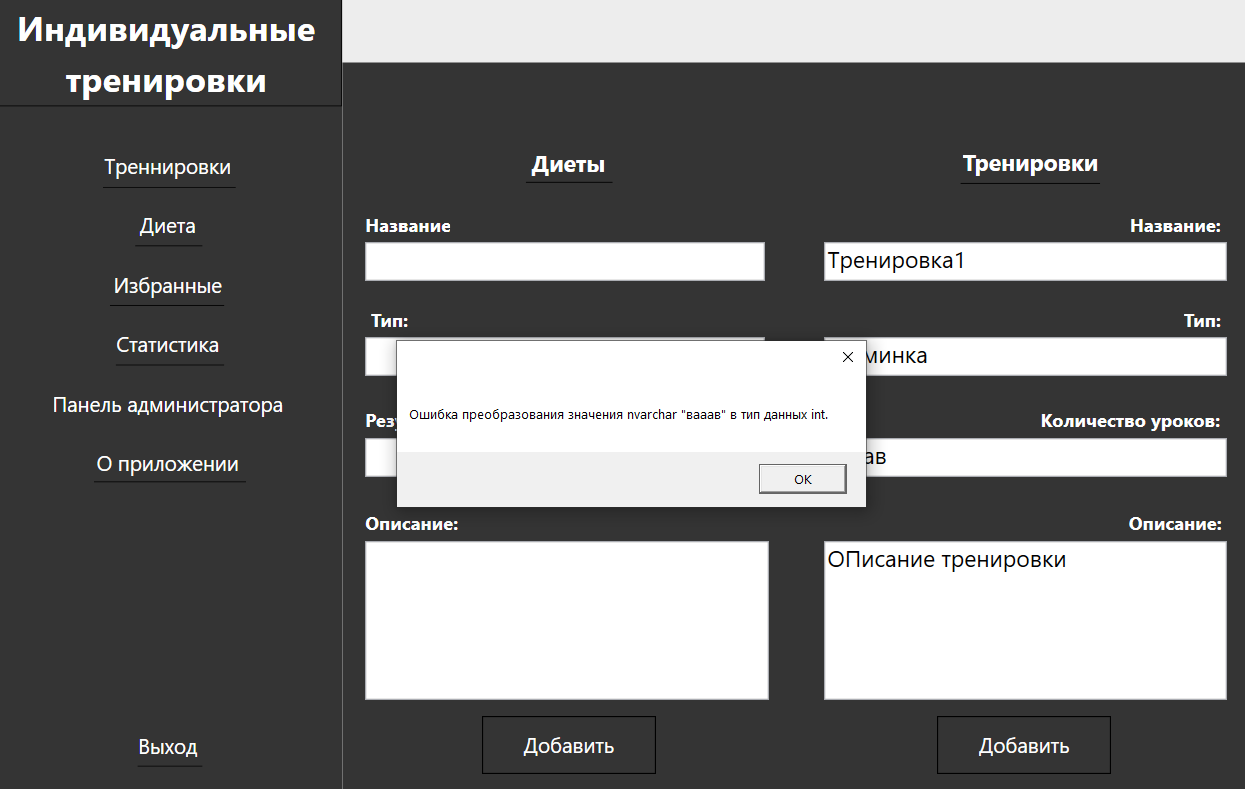


Рис. *5.2* – Ошибка неккоректности вводимых данных

Код для валидации вводимых данных приведен в **приложении Д.**

#### Руководство по использованию

#### При запуске приложения первым делом открывается окно авторизации. Мы можем зайти уже в созданный аккаунт, либо же зарегистрировать новый. Окна представлены ниже на рисунках 6.1 и 6.2. Авторизуемся под обычным пользователем.

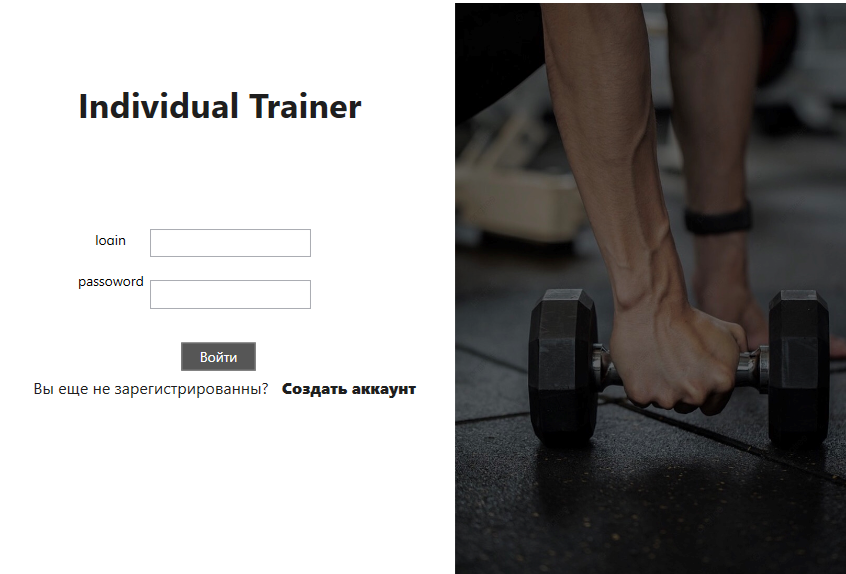


Рис. *6.1* – Окно авторизации

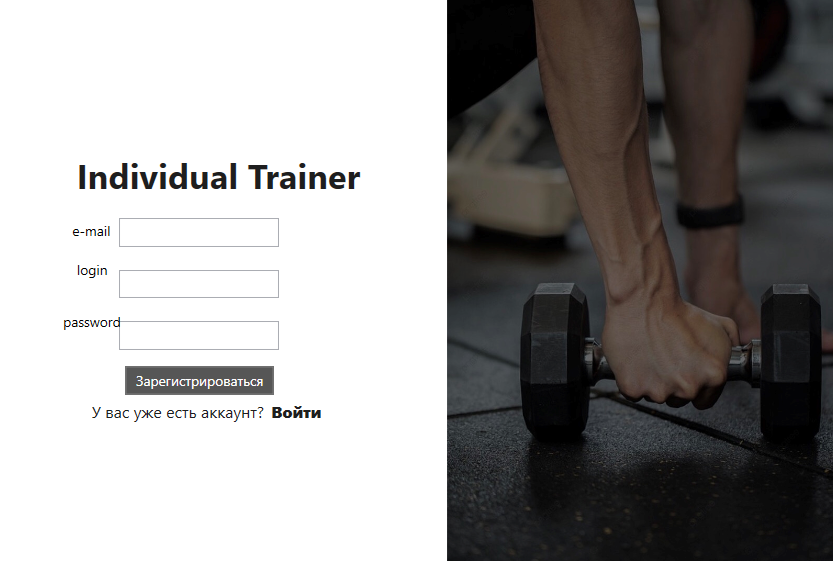
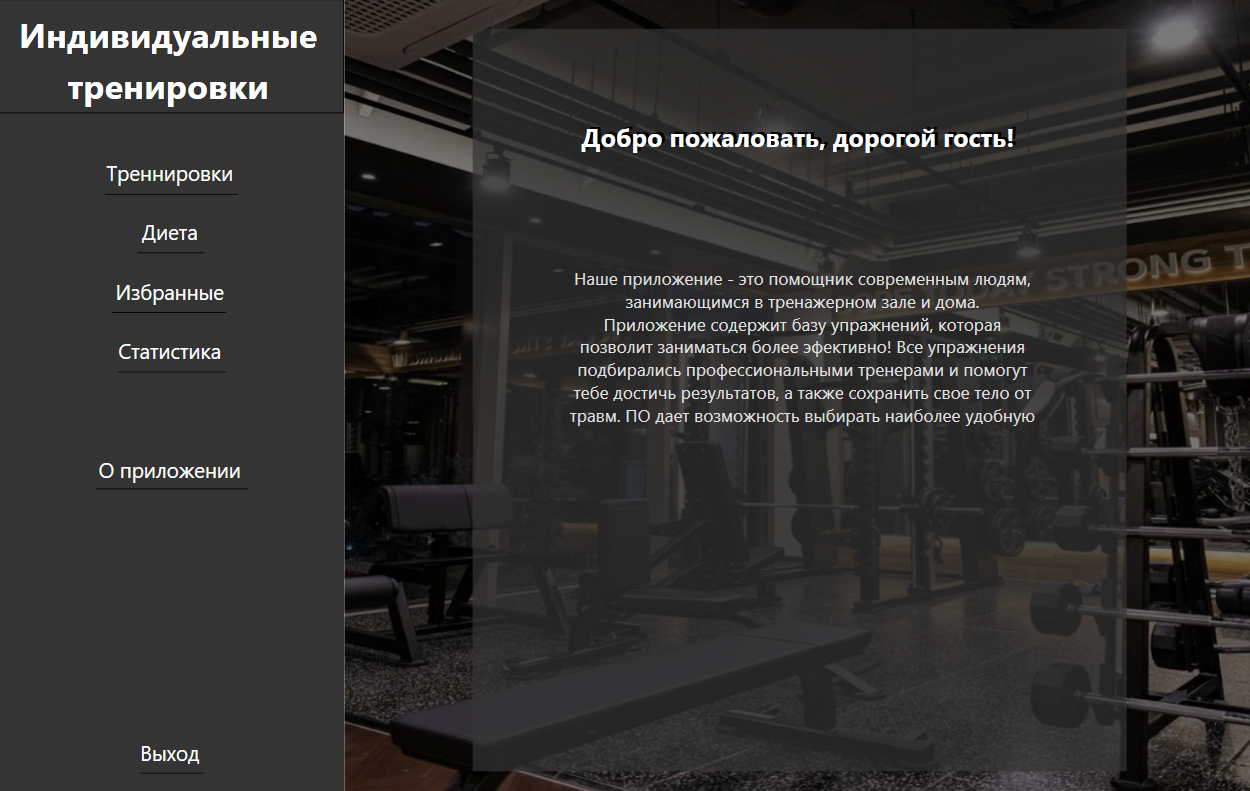


Рис. *6.2* – Окно регистрации

После авторизации мы попадаем на главную страницу, которая представлена на рисунке *6.3.*



Рис*. 6.3* – Главная страница

Далее мы можем перейти в каталог тренировок. Это представлено на рисунке 6.4

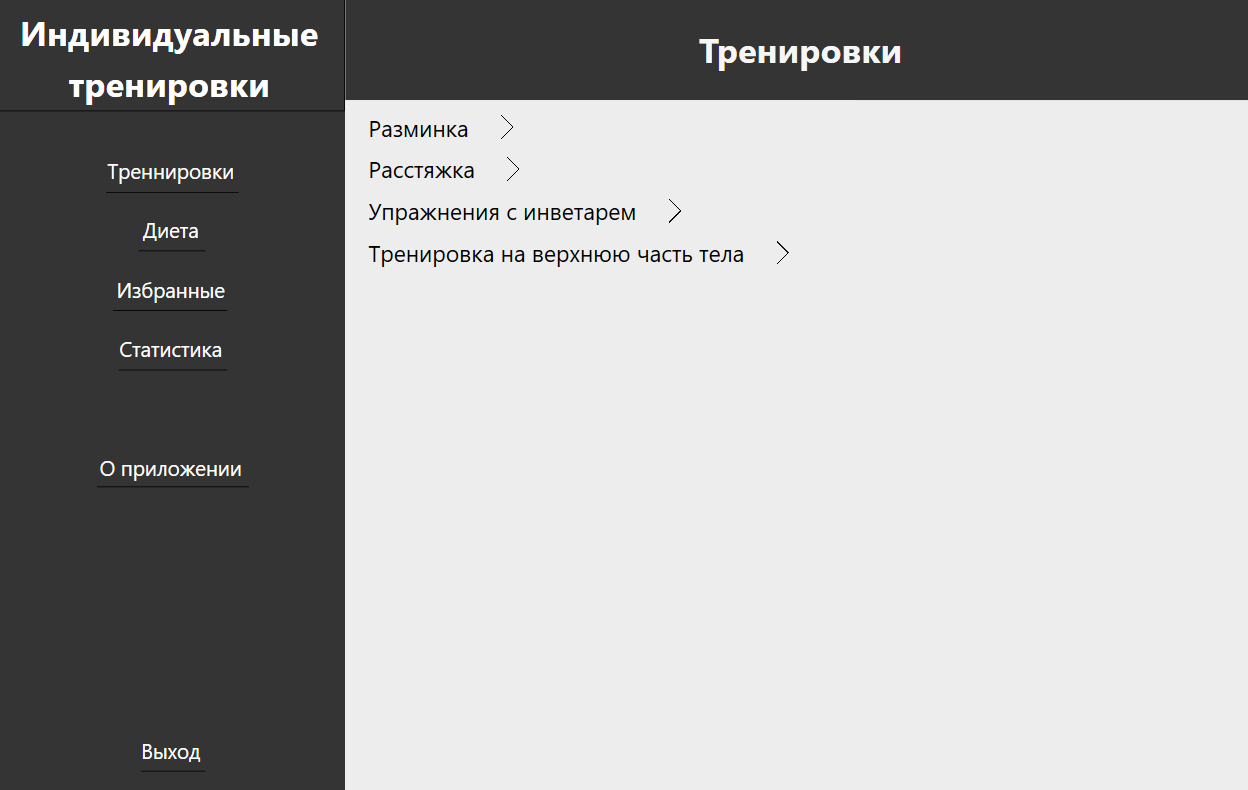


Рис. 6.4 – Каталог тренировок

После чего мы видим список предлагаемых тренировок, это представлено на рисунке *6.5.*

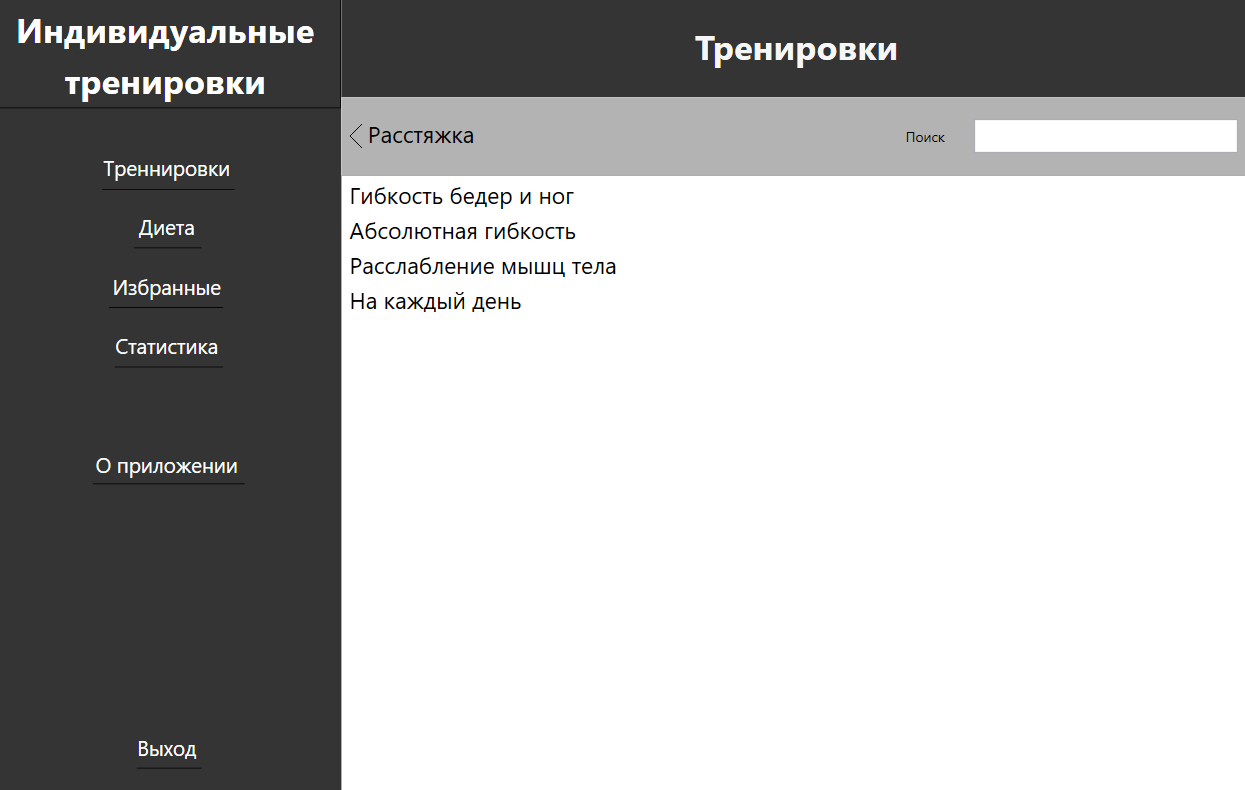


Рис. 6.5 – Подменю раздела тренировки «Разминка»

На рисунке *6.6* показано, что на выбранную тренировку можно подписаться, после чего она будет добавлена в избранное, а также отметить пройдена ли тренировка

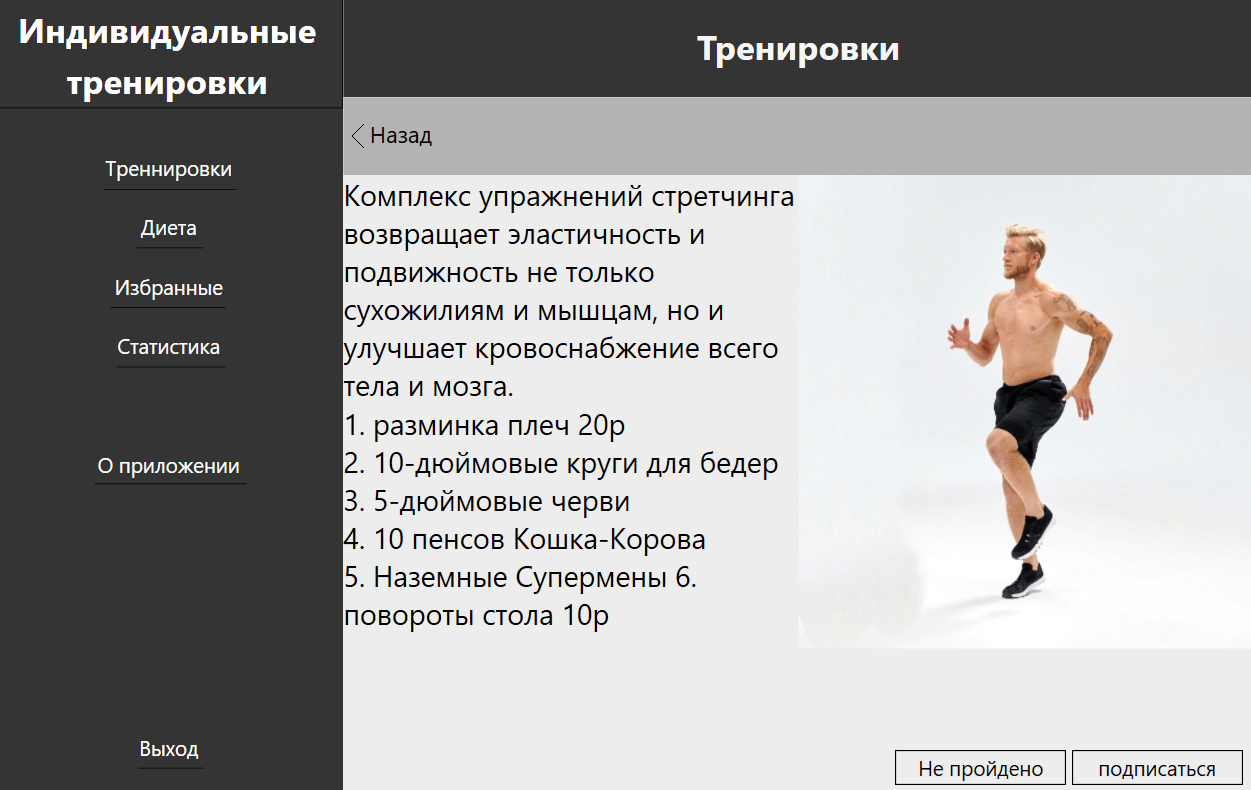


Рис. 6.6 – Выбранная тренировка

Далее можем перейти в каталог диет. Структура аналогична разделу «Тренировки». Это представлено ниже на рисунках *6.7, 6.8* и *6.9.*

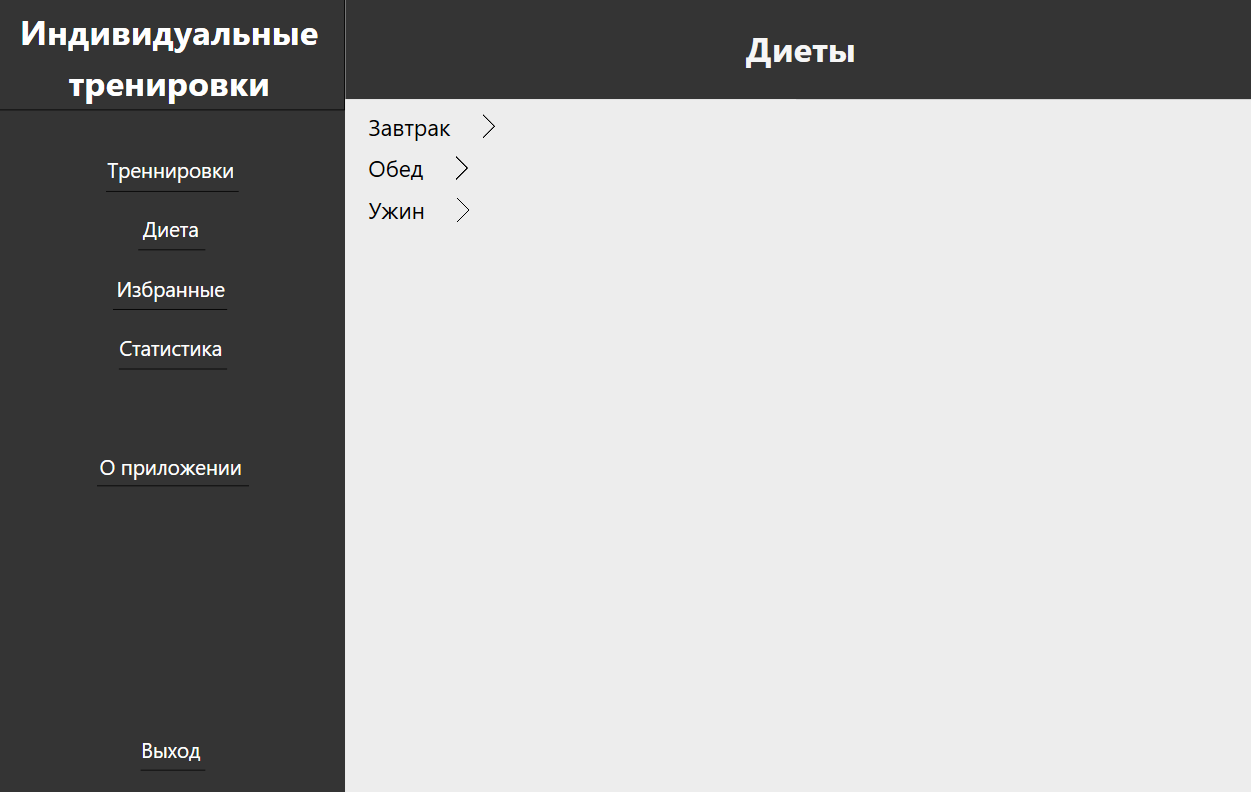


Рис. 6.7 – Каталог диет

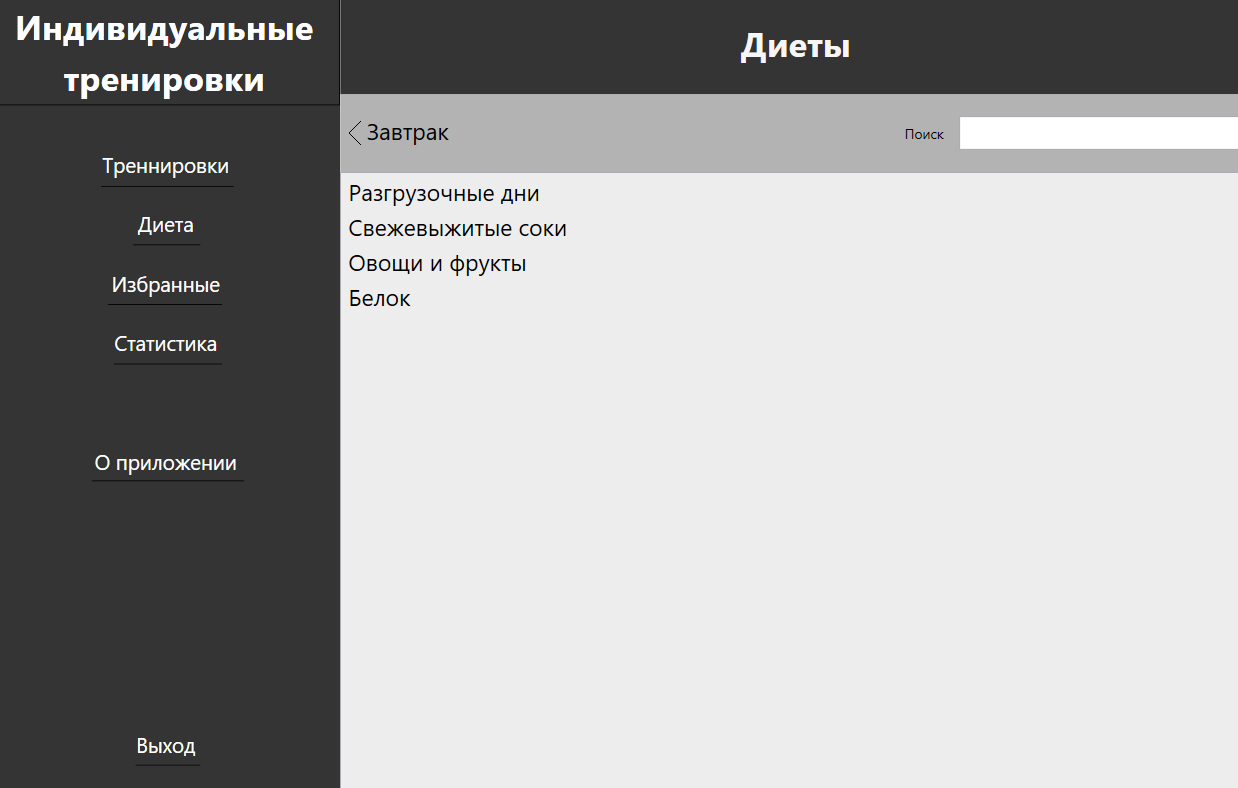


Рис. 6.8 – Подменю раздела тренировки «Завтрак»



Рис. 6.9 – Выбранная диета

Перейдя по вкладке «Избранное» будет представлен перечень всех подписок. Там можно будет отменить подписку на выбранную услугу.



Рис. 6.10 – Избранное

Также можно перейти в панель администратора (если через него авторизоваться), где мы видим перечь тренировок и диет. Там мы можем удалить выбранную тренировку/диету, а также добавить новую. Это представлено ниже на рисунках *6.11* и *6.12.*

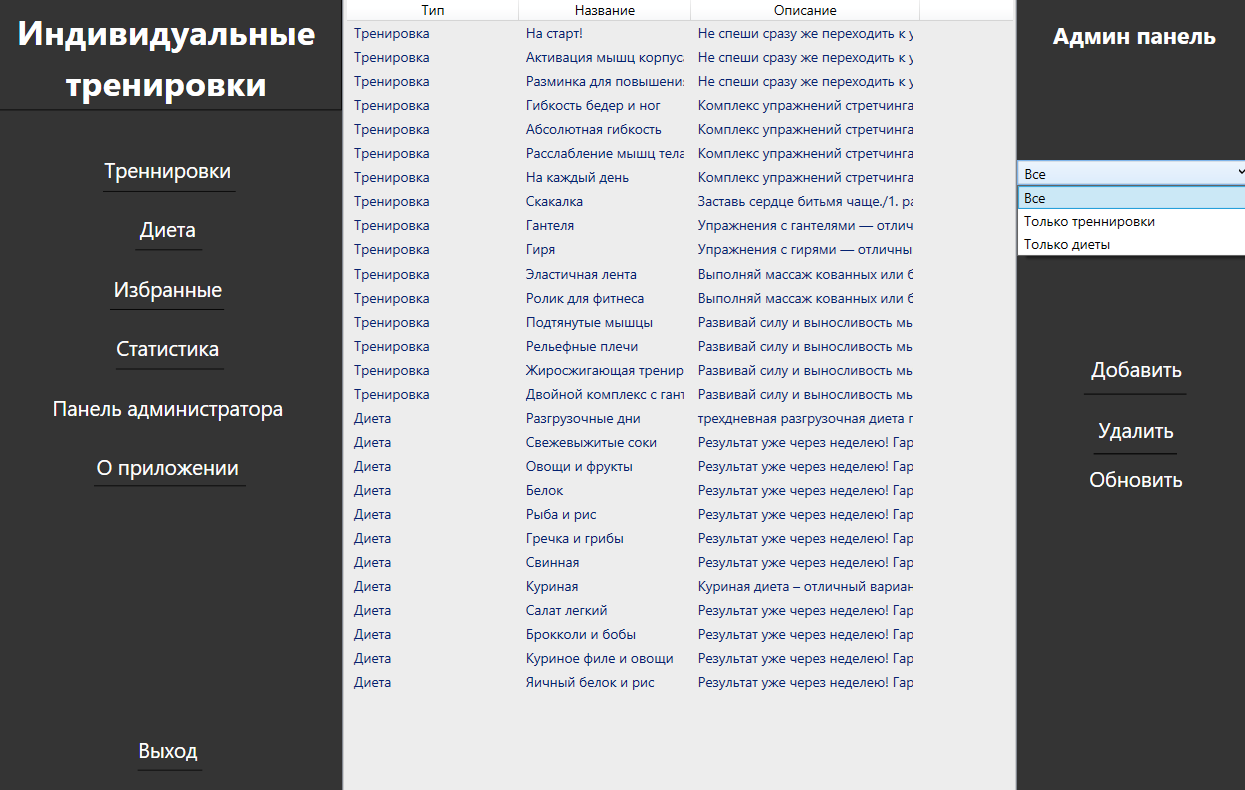


Рис. 6.11 – Панель администратора

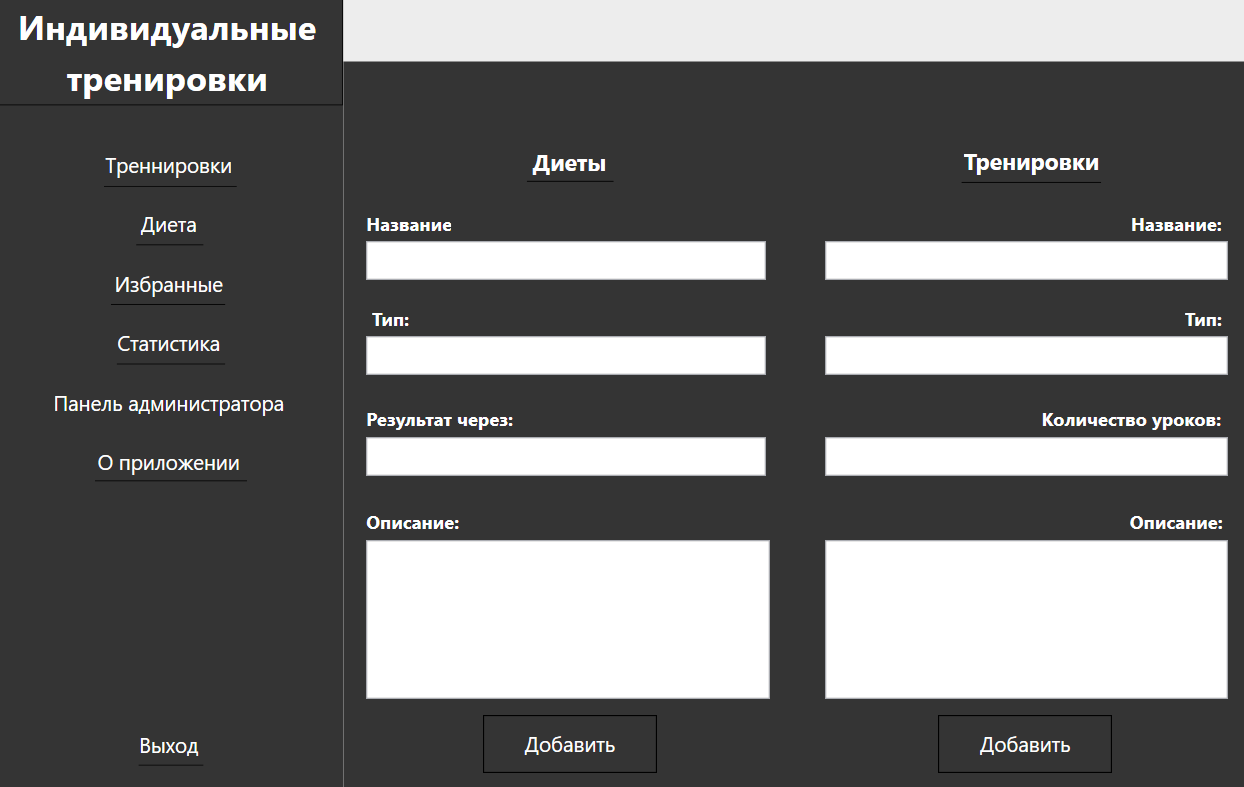


Рис. 6.12 – Добавление новой тренировки или диеты

Также можно перейти во вкладку «О приложении» и прочесть информацию.

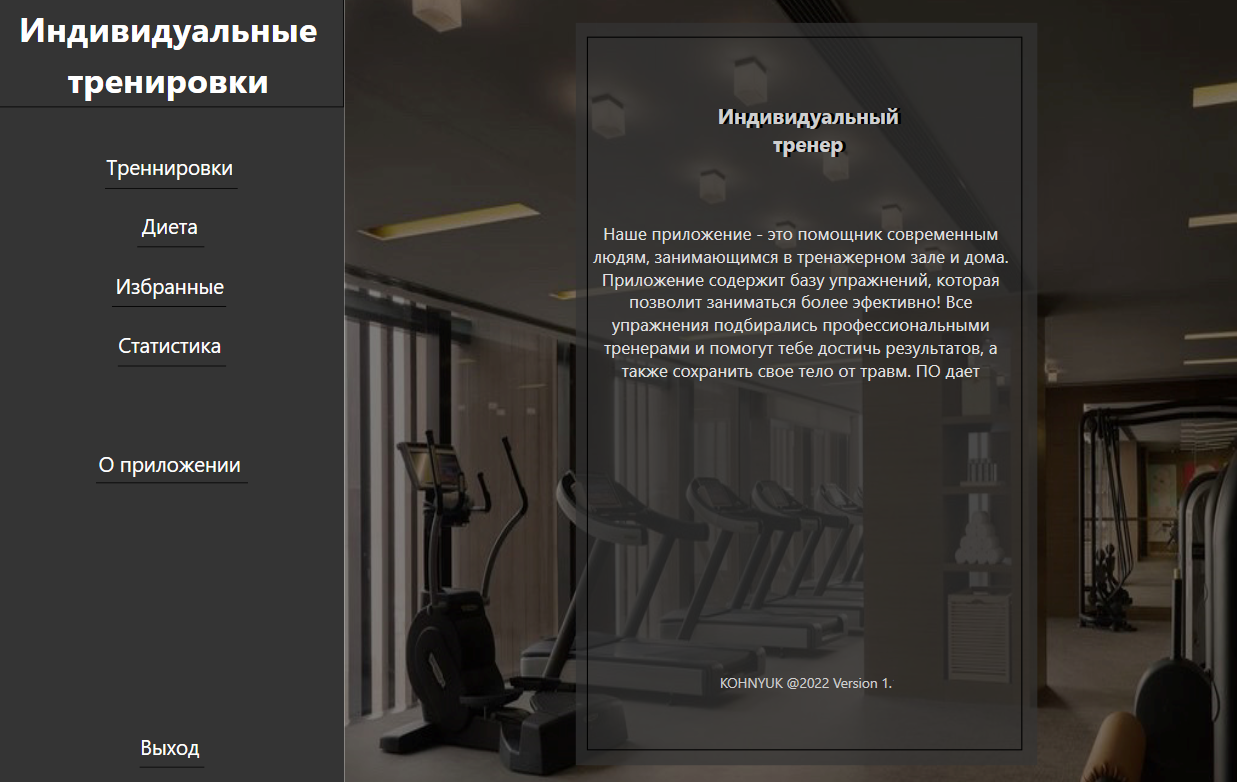


Рис. 6.8 – Страница «О приложении»

Пользователю дается возможность проанализировать свою работу, сделанную за все дни. Перед этим необходимо отмечать, пройдена ли тренировка.



Рис. 6. – Страница «Статистика»

#### ЗАКЛЮЧЕНИЕ

В данном курсовом проекте было разработано программное средство «Индивидуальный тренер» при помощи языка программирования C#, API-интерфейса Windows Presentation Foundation, технологии Entity Framework и базы данных Microsoft SQL Server. При выполнении курсового проекта использовались принципы и приемы ООП.

Программное средство предоставляет пользователю следующие функциональные возможности:

* Регистрация и авторизация
* Возможность добавить в избранное тренировку
* Возможность добавить в избранное диету
* Возможность удалить из избранного тренировку
* Возможность удалить из избранного диету
* Поиск тренировки по названию
* Поиск диеты по названию
* Отметить тренировку пройденной
* Отменить отметку о прохождении
* Сформировать отчет о проделанной работе

Администратор может:

* Добавление тренировки
* Удаление тренировки
* Добавление диеты
* Удаление диеты
* Отметить тренировку пройденной
* Отменить отметку о прохождении
* Поиск тренировки по названию
* Поиск диеты по названию
* Сформировать отчет о проделанной работе

Приложение хорошо работает с базой данных. В ходе тестирования, не было выявлено нарушений в работе приложения с базой данных.

Приложение было успешно протестировано, что означает пригодность этого приложения для своей цели.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

[1]. Пацей, Н.В. Курс лекций по языку программирования С# / Н.В. Пацей. – Минск: БГТУ, 2018. – 175 с.

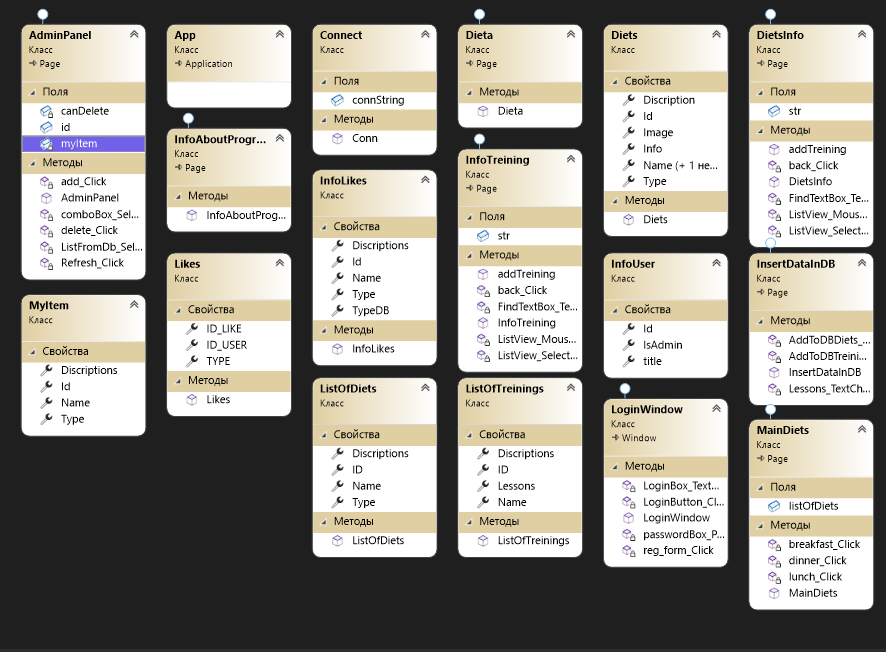
[2]. Пацей, Н.В. Технология разработки программного обеспечения / Н.В. Пацей. – Минск: БГТУ, 2016. – 129 с. [3]. MSDN сеть разработчиков в Microsoft [Электронный ресурс] / Режим доступа: http://msdn.microsoft.com/library/

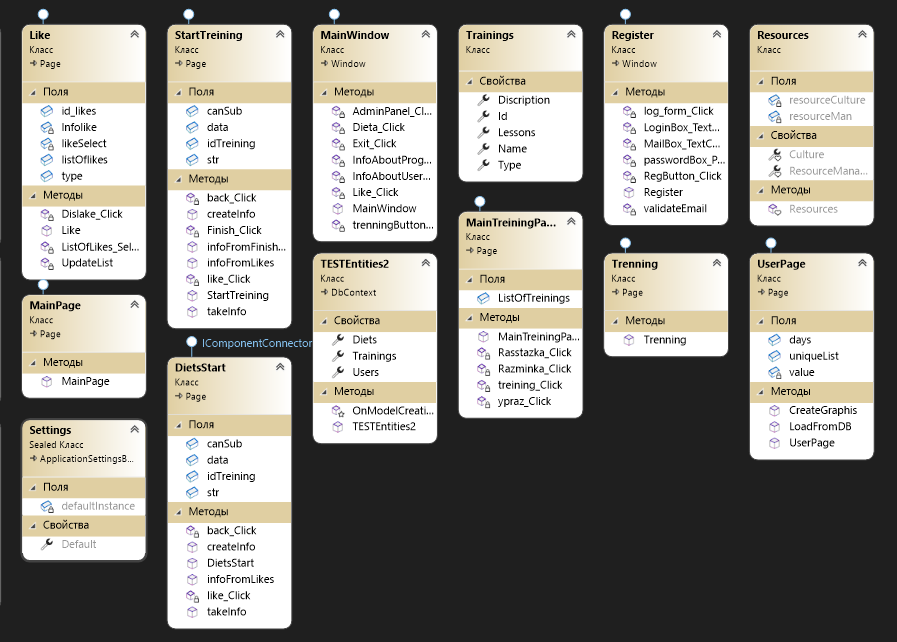
[4]. METANIT.COM Сайт о программировании [Электронный ресурс] / Режим доступа: https://metanit.com

[5]. ProfessorWeb .NET & Web Programming [Электронный ресурс] / Режим доступа: https://professorweb.ru

#### Приложение А

**Диаграмма классов**





#### Приложение Б

**Код реализации добавления тренировок и диеты**

using System;

using System.Collections.Generic;

using System.Data.SqlClient;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace Trainer.Menu.AdminSettings

{

/// <summary>

/// Логика взаимодействия для InsertDataInDB.xaml

/// </summary>

public partial class InsertDataInDB : Page

{

public InsertDataInDB()

{

InitializeComponent();

}

private void AddToDBDiets\_Click(object sender, RoutedEventArgs e)

{

try

{

ConnectToDB.Connect.Conn();

//create instanace of database connection

using (SqlConnection conn = new SqlConnection(ConnectToDB.Connect.connString))

{

conn.Open();

int id = 0;

string sql = "SELECT \*FROM Diets";

SqlCommand comm = new SqlCommand(sql, conn);

SqlDataReader reader = comm.ExecuteReader();

if (reader.HasRows) // если есть данные

{

while (reader.Read())

{

id++;

}

}

reader.Close();

sql = "INSERT INTO Diets(Name,Type,Discription) values(@Name , @Type , @Discriptions)";

SqlCommand command = new SqlCommand(sql, conn);

command.Parameters.AddWithValue("@Name", NameInDB.Text);

command.Parameters.AddWithValue("@Type", TypeInDB.Text);

command.Parameters.AddWithValue("@Discriptions", Discriptions.Text);

command.ExecuteNonQuery();

}

MessageBox.Show("Диета была добавлена");

TypeInDB.Clear();

NameInDB.Clear();

Discriptions.Clear();

}

catch (Exception exp)

{

MessageBox.Show(exp.Message);

}

}

private void AddToDBTreining\_Click(object sender, RoutedEventArgs e)

{

try

{

ConnectToDB.Connect.Conn();

//create instanace of database connection

using (SqlConnection conn = new SqlConnection(ConnectToDB.Connect.connString))

{

conn.Open();

int id = 0;

string sql = "SELECT \*FROM Trainings";

SqlCommand comm = new SqlCommand(sql, conn);

SqlDataReader reader = comm.ExecuteReader();

if (reader.HasRows) // если есть данные

{

while (reader.Read())

{

id++;

}

}

reader.Close();

sql = "INSERT INTO Trainings(Type,Name,Lessons,Discription) values(@Type , @Name,@Lessons , @Discriptions)";

SqlCommand command = new SqlCommand(sql, conn);

command.Parameters.AddWithValue("@Type", TypeInDB\_Treining.Text);

command.Parameters.AddWithValue("@Name", NameInDB\_Treining.Text);

command.Parameters.AddWithValue("@Lessons", Lessons\_Treining.Text);

command.Parameters.AddWithValue("@Discriptions", Discriptions\_Treining.Text);

command.ExecuteNonQuery();

}

MessageBox.Show("Тренировка была добавлена");

TypeInDB\_Treining.Clear();

NameInDB\_Treining.Clear();

Discriptions\_Treining.Clear();

Lessons\_Treining.Clear();

}

catch (Exception exp)

{

MessageBox.Show(exp.Message);

}

}

private void Lessons\_TextChanged(object sender, TextChangedEventArgs e)

{

}

}

}

#### Приложение В

#### Код реализации удаления тренировок или диеты

private void delete\_Click(object sender, RoutedEventArgs e)

{

if (canDelete)

{

if (myItem.Type == "Диета")

{

try

{

ConnectToDB.Connect.Conn();

//create instanace of database connection

using (SqlConnection conn = new SqlConnection(ConnectToDB.Connect.connString))

{

conn.Open();

string sql = "DELETE FROM Diets WHERE Id = @id";

SqlCommand command = new SqlCommand(sql, conn);

command.Parameters.AddWithValue("@id", id);

command.ExecuteNonQuery();

}

}

catch (Exception exp)

{

MessageBox.Show("Ошибка входа " + exp.Message);

}

}

else if(myItem.Type == "Тренировка")

{

try

{

ConnectToDB.Connect.Conn();

//create instanace of database connection

using (SqlConnection conn = new SqlConnection(ConnectToDB.Connect.connString))

{

conn.Open();

string sql = "DELETE FROM Trainings WHERE Id=@id";

SqlCommand command = new SqlCommand(sql, conn);

command.Parameters.AddWithValue("@id", id);

command.ExecuteNonQuery();

}

}

catch (Exception exp)

{

MessageBox.Show("Ошибка входа " + exp.Message);

}

}

}

canDelete = false;

#### 

#### Приложение Г

**Код валидации данных при авторизации**

using System;

using System.Collections.Generic;

using System.Data.SqlClient;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace Trainer.Login\_Register

{

/// <summary>

/// Логика взаимодействия для LoginWindow.xaml

/// </summary>

public partial class LoginWindow : Window

{

public LoginWindow()

{

InitializeComponent();

}

private void LoginButton\_Click(object sender, RoutedEventArgs e)

{

try

{

if (!String.IsNullOrEmpty(LoginBox.Text) && !String.IsNullOrEmpty(passwordBox.Password))//пустая ли строка

{

ConnectToDB.Connect.Conn();//путь к файлу с бд

//create instanace of database connection

using (SqlConnection conn = new SqlConnection(ConnectToDB.Connect.connString))

{

int id = 0;

bool canEntry = false;

conn.Open();

string sql = "SELECT \* FROM Users";

SqlCommand command = new SqlCommand(sql, conn);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

if (reader.GetValue(1).ToString() == LoginBox.Text && reader.GetValue(2).ToString() == passwordBox.Password)

{

InfoUser.Id = Convert.ToInt32(reader.GetValue(0));

InfoUser.IsAdmin = Convert.ToInt32(reader.GetValue(5));

canEntry = true;

MainWindow main = new MainWindow();

main.Show();

this.Close();

break;

}

else

{

canEntry = false;

}

}

if (!canEntry)

{

label\_error.Content = "Пользователя нет в системе";

}

}

}

else

{

label\_error.Content = "Заполните все поля";

LoginBox.Foreground = Brushes.Red;

passwordBox.Foreground = Brushes.Red;

}

}catch(Exception exp)

{

MessageBox.Show("Ошибка входа "+exp.Message);

}

}

private void reg\_form\_Click(object sender, RoutedEventArgs e)

{

Register register = new Register();

register.Show();

this.Close();

}

private void LoginBox\_TextChanged(object sender, TextChangedEventArgs e)

{

if (!System.Text.RegularExpressions.Regex.IsMatch(LoginBox.Text, "^[a-zA-Z0-9]") && LoginBox.Text.Length != 0)

{

LoginBox.Foreground = Brushes.Red;

label\_error.Content = "Запрещена кириллица";

}

else

{

LoginBox.Foreground = Brushes.Black;

label\_error.Content = "";

}

}

private void passwordBox\_PasswordChanged(object sender, RoutedEventArgs e)

{

if (!System.Text.RegularExpressions.Regex.IsMatch(passwordBox.Password, "^[a-zA-Z0-9]") && passwordBox.Password.Length != 0)

{

passwordBox.Foreground = Brushes.Red;

label\_error.Content = "Запрещена кириллица";

}

else

{

passwordBox.Foreground = Brushes.Black;

label\_error.Content = "";

}

}

}

}

#### Приложение Д

**Код валидации данных при добавления тренировки или диеты**

using System;

using System.Collections.Generic;

using System.Data.SqlClient;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace Trainer.Menu.AdminSettings

{

/// <summary>

/// Логика взаимодействия для InsertDataInDB.xaml

/// </summary>

public partial class InsertDataInDB : Page

{

public InsertDataInDB()

{

InitializeComponent();

}

private void AddToDBDiets\_Click(object sender, RoutedEventArgs e)

{

try

{

ConnectToDB.Connect.Conn();

//create instanace of database connection

using (SqlConnection conn = new SqlConnection(ConnectToDB.Connect.connString))

{

conn.Open();

int id = 0;

string sql = "SELECT \*FROM Diets";

SqlCommand comm = new SqlCommand(sql, conn);

SqlDataReader reader = comm.ExecuteReader();

if (reader.HasRows) // если есть данные

{

while (reader.Read())

{

id++;

}

}

reader.Close();

sql = "INSERT INTO Diets(Name,Type,Discription) values(@Name , @Type , @Discriptions)";

SqlCommand command = new SqlCommand(sql, conn);

command.Parameters.AddWithValue("@Name", NameInDB.Text);

command.Parameters.AddWithValue("@Type", TypeInDB.Text);

command.Parameters.AddWithValue("@Discriptions", Discriptions.Text);

command.ExecuteNonQuery();

}

MessageBox.Show("Диета была добавлена");

TypeInDB.Clear();

NameInDB.Clear();

Discriptions.Clear();

}

catch (Exception exp)

{

MessageBox.Show(exp.Message);

}

}

private void AddToDBTreining\_Click(object sender, RoutedEventArgs e)

{

try

{

ConnectToDB.Connect.Conn();

//create instanace of database connection

using (SqlConnection conn = new SqlConnection(ConnectToDB.Connect.connString))

{

conn.Open();

int id = 0;

string sql = "SELECT \*FROM Trainings";

SqlCommand comm = new SqlCommand(sql, conn);

SqlDataReader reader = comm.ExecuteReader();

if (reader.HasRows) // если есть данные

{

while (reader.Read())

{

id++;

}

}

reader.Close();

sql = "INSERT INTO Trainings(Type,Name,Lessons,Discription) values(@Type , @Name,@Lessons , @Discriptions)";

SqlCommand command = new SqlCommand(sql, conn);

command.Parameters.AddWithValue("@Type", TypeInDB\_Treining.Text);

command.Parameters.AddWithValue("@Name", NameInDB\_Treining.Text);

command.Parameters.AddWithValue("@Lessons", Lessons\_Treining.Text);

command.Parameters.AddWithValue("@Discriptions", Discriptions\_Treining.Text);

command.ExecuteNonQuery();

}

MessageBox.Show("Тренировка была добавлена");

TypeInDB\_Treining.Clear();

NameInDB\_Treining.Clear();

Discriptions\_Treining.Clear();

Lessons\_Treining.Clear();

}

catch (Exception exp)

{

MessageBox.Show(exp.Message);

}

}

private void Lessons\_TextChanged(object sender, TextChangedEventArgs e)

{

}

}

}