Министерство образования и науки Республики Башкортостан

Государственное автономное профессиональное образовательное учреждение

Уфимский колледж статистики, информатики и вычислительной техники

|  |  |
| --- | --- |
|  | УТВЕРЖДАЮ  Заместитель директора  по учебной работе  З.З. Курмашева  « » 2025г. |

ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ И РАЗРАБОТКА ПРИЛОЖЕНИЯ

ДЛЯ ОРГАНИЗАЦИИ ДЕЯТЕЛЬНОСТИ АЛЬПИНИСТСКОГО КЛУБА

Пояснительная записка к курсовому проекту

МДК.11.01 Технология разработки и защиты баз данных

|  |  |
| --- | --- |
|  | Руководитель проекта  Р.Ф.Каримова  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2025 г. |
|  | Студент гр. 22П-1  Л.В.Иванова  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2025 г. |

2025

Министерство образования и науки Республики Башкортостан

Государственное автономное профессиональное образовательное учреждение

Уфимский колледж статистики, информатики и вычислительной техники

|  |  |
| --- | --- |
|  | УТВЕРЖДАЮ  Заместитель директора  по учебной работе  З.З. Курмашева  « » 2025 г. |

ЗАДАНИЕ

на курсовой проект студенту дневного отделения, группы 22П-1, специальности 09.02.07 Информационные системы и программирование.

Фамилия, имя, отчество: Иванова Любовь Вячеславовна

Тема курсового проекта: «Проектирование базы данных и разработка приложения для организации деятельности альпинистского клуба».

Текст задания:

при выполнении курсового проекта должны быть решены следующие задачи:

1. спроектирована база данных;
2. разработана и реализована структура программы;
3. реализованы функции регистрации и управления учетными записями участников, поиска информации по различным критериям, формирования отчетов и статистики, управления заявками на восхождения, поддержки личных кабинетов участников, экспорта данных в определенном формате, ведение базы данных по горам, восхождениям и участникам;

В результате выполнения курсового проекта должны быть представлены:

1. пояснительная записка, состоящая из следующих разделов:

Введение

1 Постановка задачи

2 Экспериментальный раздел

Заключение

Приложения

Список сокращений

Список использованных источников

1. электронный носитель, содержащий разработанный программный продукт;
2. презентация курсового проекта в электронном виде.

Список рекомендуемых источников:

1 ГОСТ 34.321-96 Информационные технологии. Система стандартов по базам данных. Эталонная модель управления данными: межгосударственный стандарт: дата введения 2001-07-01 - https://internetlaw.ru/gosts/gost/6808/?ysclid=led2xw9tta320329480 (дата обращения 18.02.2023) – Текст: электронный.

2 Гордеев, С. И. Организация баз данных в 2 ч. Часть 1: учебник для среднего профессионального образования / С. И. Гордеев, В. Н. Волошина. — 2-е изд., испр. и доп. — Москва: Издательство Юрайт, 2023. — 310 с. — (Профессиональное образование). — ISBN 978-5-534-11626-7. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/518510 (дата обращения: 14.03.2023).

3 Гордеев, С. И. Организация баз данных в 2 ч. Часть 2: учебник для вузов / С. И. Гордеев, В. Н. Волошина. — 2-е изд., испр. и доп. — Москва: Издательство Юрайт, 2023. — 513 с. — (Высшее образование). — ISBN 978-5-534-04470-6. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/515097 (дата обращения: 14.03.2023).

4 Прохоренок, Н.А. HTML, JavaScript, PHP и MySQL. Джентльменский набор Web-мастера / Н. А. Прохоренок. – 2-е изд. – Санкт-Петербург: БХВ-Петербург, 2019. – 880 с. – (Профессиональное программирование: PRO). – ISBN 978-5-9775-0396-9. Текст: непосредственный.

Задание к выполнению получил «29» января 2025 г.

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Иванова Любовь Вячеславовна

Срок окончания «25» июня 2025 г.

Руководитель курсового проекта Р.Ф. Каримова

Задание рассмотрено на заседании цикловой комиссии информатики

протокол № 4 от «11» января 2025 г.

Председатель цикловой комиссии информатики О.В.Фатхулова

Министерство образования и науки Республики Башкортостан

Государственное автономное профессиональное образовательное учреждение

Уфимский колледж статистики, информатики и вычислительной техники

ЗАКЛЮЧЕНИЕ

на курсовой проект

Студент Иванова Любовь Вячеславовна

Группа 22П-1

Специальность 09.02.07 Информационные системы и программирование

Тема Проектирование базы данных и разработка приложения для организации деятельности альпинистского клуба

Объем курсового проекта:

количество листов пояснительной записки

количество листов графической части

Заключение о степени соответствия заданию на курсовое проектирование

Характеристика качеств, проявленных студентом при работе над проектом: самостоятельность, дисциплинированность, умение планировать работу и пользоваться литературным материалом и т.д.

Положительные стороны курсового проекта

Недостатки курсового проекта

Характеристика общетехнической и специальной подготовки студента

Заключение и предлагаемая оценка за курсовой проект

Руководитель курсового проекта \_\_\_\_\_\_\_\_\_Р.Ф. Каримова

«\_\_\_\_» 2025 г.

Подпись

АННОТАЦИЯ

Пояснительная записка к курсовому проекту содержит постановку и программу решения задачи «Проектирование базы данных и разработка приложения для организации деятельности альпинистского клуба».

Программа MountaineeringClub.exe написана на языке C# в среде программирования Visual Studio 2022 с использованием системы управления базой данных MySQL, предназначена для работы в операционной системе MS Windows 10 и выше, отлажена на данных контрольного примера.

СОДЕРЖАНИЕ

|  |  |
| --- | --- |
|  | лист |
| Введение | 4 |
| 1 Постановка задачи | 6 |
| 1.1 Описание предметной области | 6 |
| 1.2 Описание входной информации | 8 |
| 1.3 Описание выходной информации | 8 |
| 1.4 Концептуальное моделирование | 9 |
| 1.5 Логическое моделирование | 10 |
| 1.6 Описание структуры базы данных | 12 |
| 1.7 Контрольный пример | 14 |
| 1.8 Общие требования к программному продукту | 14 |
| 2 Экспериментальный раздел | 16 |
| 2.1 Описание программы | 16 |
| 2.2 Протокол тестирования программного продукта | 25 |
| 2.3 Руководство пользователя | 34 |
| 2.4 Меры по обеспечению защиты информации | 44 |
| Заключение | 46 |
| Приложения | 47 |
| Список сокращений | 106 |
| Список использованных источников | 107 |

ВВЕДЕНИЕ

Альпинистские клубы играют важную роль в объединении энтузиастов, для которых горные восхождения – не просто спорт, а философия жизни. Их деятельность включает организацию экспедиций, обучение технике безопасности, поддержание снаряжения, ведение архивов восхождений и популяризацию альпинизма. Однако управление такими клубами связано с рядом сложностей: ручной учет участников, планирование мероприятий, обработка заявок и анализ статистики требуют значительных временных затрат.

Актуальность работы обусловлена отсутствием специализированных IT-решений, полностью адаптированных под нужды альпинистских сообществ. Анализ существующих программных продуктов показал, что большинство из них либо слишком универсальны, либо узконаправленны, не охватывая всех аспектов клубной работы. Это приводит к фрагментации данных: информация хранится в разных сервисах, что усложняет ее обработку и повышает риск ошибок. Внедрение автоматизированной системы, учитывающей специфику альпинизма, позволит оптимизировать административные процессы, повысить прозрачность деятельности клуба.

Цель проекта – разработка веб-приложения для альпинистского клуба, обеспечивающего:

- централизованный учет участников, гор и восхождений;

- подачу и обработку заявок на экспедиции;

- формирование рейтингов альпинистов и статистики по маршрутам;

- генерацию отчетов и экспорт данных в стандартизированных форматах.

Задачами курсового проекта являются:

* описать предметную область;
* разработать структуру базы данных;
* разработать приложение;
* провести тестирование приложения.

1 Постановка задачи

* 1. Описание предметной области

Требуется разработать информационную систему для автоматизации учета восхождений в альпинистском клубе, хранения информации об участниках, горах, восхождениях, возможности регистрации пользователей, получения доступа к личному кабинету, управления данными.

Данная информационная система предполагает наличие трех групп пользователей: гость, участник и администратор.

Гость имеет доступ к функциям:

* просмотр общей информации о клубе, информации о горах, восхождениях.

Участник имеет доступ к функциям:

* личный кабинет:
* просмотр личного профиля;
* редактирование личной информации;
* редактирование статуса (покидание клуба);
* подача заявки на новое восхождение с указанием роли (участник/руководитель) и отслеживание статуса заявок.

Администратор имеет следующие возможности:

* управление восхождениями: изменения статуса восхождений (запланировано/не удалось/отменено/выполнено)
* управление пользователями: изменение статуса участников (состоит/не состоит в клубе);
* генерация отчетов с информацией об участниках, о горах и об восхождениях.

В базе данных должны храниться следующие справочники: участники, горы, восхождения.

После регистрации участника в клубе система фиксирует его персональные данные и создает учетную запись. Участник может подавать заявки на восхождения, где указывает желаемую роль (участник или руководитель). Каждое восхождение имеет определенный маршрут. Если заявка одобрена, участник включается в состав группы. После завершения восхождения фиксируются его результаты и обновляется рейтинг участника.

Таблица «Пользователь» необходима для хранения всех групп пользователей. В таблице содержится следующая информация:

* идентификатор пользователя;
* ФИО (фамилия, имя, отчество);
* контактный телефон;
* адрес электронной почты;
* пароль для доступа;
* дата рождения;
* дата регистрации;
* роль(администратор/зарегистрированный участник).

В таблице «Участник» содержится:

* идентификатор участника;
* идентификатор пользователя;
* статус членства(активный/неактивный);
* история восхождений.

Каждый участник может иметь несколько заявок на восхождения. В таблице «Заявка» содержится:

* идентификатор заявки;
* идентификатор участника;
* дата подачи заявки;
* желаемая роль (участник/руководитель);
* статус заявки (на рассмотрении/одобрено/отклонено/выполнено).
* идентификатор восхождения;

Восхождения организуются на различные горы. В таблице «Гора» содержится:

* идентификатор горы;
* название;
* высота;
* страна;
* регион;
* описание.

Каждое восхождение имеет свои характеристики. В таблице «Восхождение» содержится:

* идентификатор восхождения;
* идентификатор горы;
* дата начала;
* дата окончания;
* описание маршрута;
* статус восхождения (запланировано/не удалось/отменено/выполнено);
* целевая точка.

Восхождение может иметь фотографии. В таблице «Фотография» содержится:

* идентификатор фотографии;
* идентификатор восхождения;
* изображение;
* описание.

Входными данными являются сведения об участниках клуба, горах и планируемых восхождениях.

Выходными данными являются отчеты о восхождениях, рейтинги участников и экспортируемые документы.

1.2 Описание входной информации

Входной информацией для выполнения задачи являются справочники: участники, горы, восхождения, заявки на участие, критерии поиска.

1.3 Описание выходной информации

Выходной информацией является отчетная документация:

* отчеты о восхождениях: списки участников, даты, маршруты;
* рейтинги участников: количество восхождений;
* поисковые результаты: информация о горах, участниках и восхождениях по заданным критериям.

Описание выходных документов представлено в таблице 1.3.1.

Таблица 1.3.1 – Описание выходных документов

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Наименование документа (шифр) | Периодичность выдачи документа | Кол-во экз. | Куда передаются | Поля сортировки | Поля группировки | Итоги |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Отчет по завершенным восхождениям | По мере необходимости | 2 | Администратору, Участникам | По дате восхождения | - | - |
| Рейтинг участников | По мере необходимости | 1 | Администратору, Участникам | По количеству восхождений | - | - |

Продолжение таблицы 1.3.1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Статистика по горам | По мере необходимости | 1 | Администратору | По количеству восхождений | - | - |

Шаблоны выходных документов представлены в приложении А.

1.4. Концептуальное моделирование

Концептуальная модель базы данных - это некая наглядная диаграмма, нарисованная в принятых обозначениях и подробно показывающая связь между объектами и их характеристиками. Создается концептуальная модель для дальнейшего проектирования базы данных и перевод ее, например, в реляционную базу данных. На концептуальной модели в визуально удобном виде прописываются связи между объектами данных и их характеристиками.

В концептуальной модели есть принятые обозначения элементов. Сущность или объект обозначать прямоугольником, отношения обозначать ромбом, атрибуты объектов, обозначаются овалом. Если сущность связана с отношением, то их связь обозначается прямой линией со стрелкой. Необязательная связь обозначается пунктирной линией. Мощная связь обозначается двойной линией.

Концептуальная модель базы данных представлена в рисунке 1.4.1

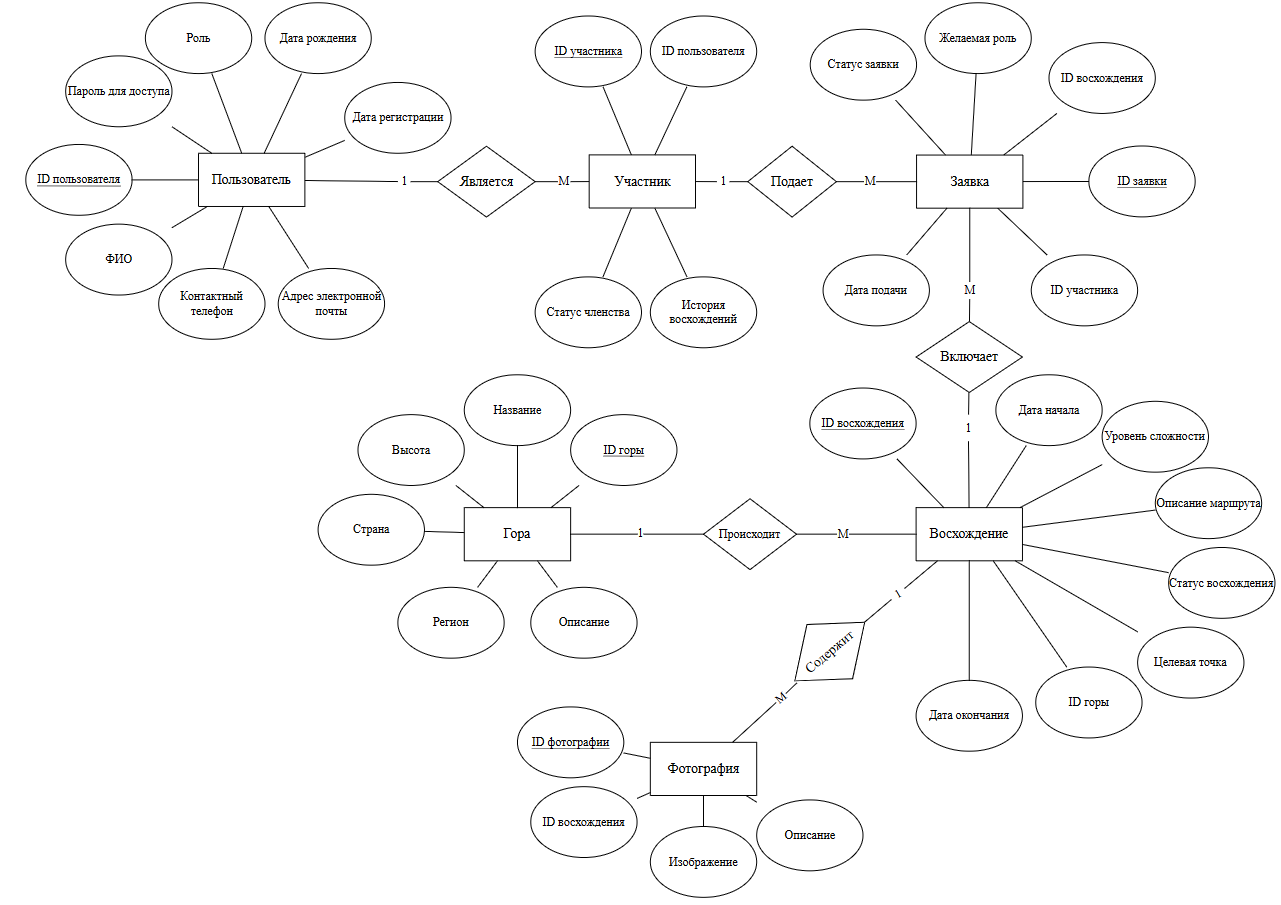


Рисунок 1.4.1 – Концептуальная модель БД

1.5. Логическое моделирование

При логическом моделировании происходит окончательное определение структуры данных, определяются ограничения, накладываемые на эти данные, целью которых является обеспечить целостность данных. Наиболее распространенной моделью данных является реляционная модель. В этой модели данных каждая сущность представляется в виде таблицы.

Логическое моделирование заключается в переходе от концептуальной модели к взаимосвязанным таблицам. Этот переход состоит из следующих шагов:

1. Преобразование сущностей:

- каждая простая сущность становится таблицей.

- каждый атрибут становится столбцом таблицы.

- уникальный идентификатор сущности становится ключом таблицы.

2. Преобразование связи:

- сущности, связанные обязательной связью один к одному можно объединить в одну таблицу.

- связи типа один к одному возможные и связи типа один ко многим реализуются путем переноса ключевых атрибутов таблиц, соответствующих сущностей, стоящих со стороны один в таблице соответствующих сущностей, стоящих со стороны многие.

- связи типа многие ко многим реализуются при помощи промежуточных таблиц, содержащих ключевые атрибуты связываемых таблиц в качестве внешних ключей.

Схема данных – это структура базы данных, описанная на формальном языке, поддерживаемом СУБД (системой управления базы данных). В реляционных базах данных схема определяет таблицы, поля в каждой таблице и ограничения целостности, такие как первичный и внешний ключи.

Схема данных представлена на рисунке 1.5.1.

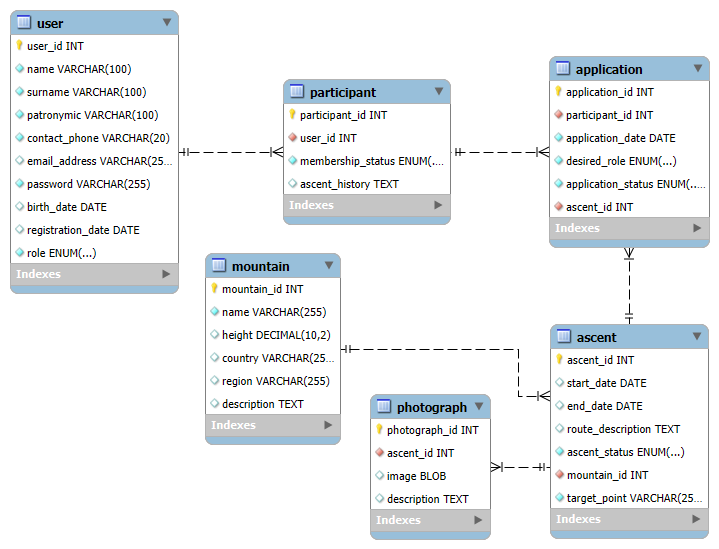


Рисунок 1.5.1 – Схема БД

1.6. Описание структуры базы данных

Описание структуры базы данных представлено в таблицах 1.6.1 – 1.6.6.

Таблица 1.6.1 - User (Пользователь)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип, длина | Примечания |
| 1 | 2 | 3 | 4 |
| ID пользователя | user\_id | INT | Первичный ключ,  Обязательное поле, Автоинкремент |
| Имя пользователя | name | VARCHAR(100) | Обязательное поле |

Продолжение таблицы 1.6.1

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| Фамилия пользователя | surname | VARCHAR(100) | Обязательное поле |
| Отчество пользователя | patronymic | VARCHAR(100) | Обязательное поле |
| Контактный телефон | contact\_phone | VARCHAR(20) | Обязательное поле, Уникальное поле |
| Адрес электронной почты | email\_address | VARCHAR(255) | Уникальное поле |
| Пароль для доступа | password | VARCHAR(255) | Обязательное поле |
| Дата рождения | birth\_date | DATE |  |
| Дата регистрации | registration\_date | DATE |  |
| Роль пользователя | role | ENUM('администратор', 'гость', 'зарегистрированный участник') | Обязательное поле |

Таблица 1.6.2 - Participant (Участник)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип, длина | Примечания |
| 1 | 2 | 3 | 4 |
| ID участника | participant\_id | INT | Первичный ключ,  Обязательное поле, Автоинкремент |
| ID пользователя | user\_id | INT | Обязательное поле, Внешний ключ |

Продолжение таблицы 1.6.2

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| Статус членства | membership\_status | ENUM('активный', неактивный) | Обязательное поле |
| История восхождений | ascent\_history | TEXT |  |

Таблица 1.6.3 - Application (Заявка)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип, длина | Примечания |
| ID заявки | application\_id | INT | Первичный ключ, Обязательное поле, Автоинкремент |
| ID участника | participant\_id | INT | Обязательное поле, Внешний ключ |
| Дата подачи | application\_date | DATE | Обязательное поле |
| Желаемая роль | desired\_role | ENUM('участник', 'руководитель') | Обязательное поле |
| Статус заявки | application\_status | VARCHAR(50) | Обязательное поле |
| ID восхождения | ascent\_id | INT | Обязательное поле, Внешний ключ |

Таблица 1.6.4 - Mountain (Гора)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип, длина | Примечания |
| 1 | 2 | 3 | 4 |
| ID горы | mountain\_id | INT | Первичный ключ, Обязательное поле, Автоинкремент |

Продолжение таблицы 1.6.4

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| Название | name | VARCHAR(255) | Обязательное поле |
| Высота | height | DECIMAL(10, 2) |  |
| Страна | country | VARCHAR(255) |  |
| Регион | region | VARCHAR(255) |  |
| Описание | description | TEXT |  |

Таблица 1.6.5 - Ascent (Восхождение)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип, длина | Примечания |
| ID восхождения | ascent\_id | INT | Первичный ключ, Обязательное поле, Автоинкремент |
| Дата начала | start\_date | DATE |  |
| Дата окончания | end\_date | DATE |  |
| Описание маршрута | route\_description | TEXT |  |
| Статус восхожденияя | ascent\_status | ENUM('запланировано', 'не удалось', 'отменено', 'выполнено') | Обязательное поле |
| ID горы | mountain\_id | INT | Внешний ключ, Обязательное поле |
| Целевая точка | target\_point | VARCHAR(255) | Обязательное поле |

Таблица 1.6.6 - Photograph (Фотография)

|  |  |  |  |
| --- | --- | --- | --- |
| Содержание поля | Имя поля | Тип, длина | Примечания |
| ID фотографии | photograph\_id | INT | Первичный ключ, Обязательное поле, Автоинкремент |
| ID восхождения | ascent\_id | INT | Обязательное поле, Внешний ключ |
| Изображение | image | BLOB | Обязательное поле |
| Описание | description | TEXT |  |

1.7. Контрольный пример

Контрольный пример является ручным подсчётом задачи. По составленной программе обрабатываются исходные данные контрольного примера. Полученные результаты сравниваются с известными результатами контрольного примера. При несовпадении результатов производится поиск, исправление ошибок, и снова производится выполнение программы.

Входная информация контрольных примеров представлена в приложении Б.

Выходные данные для контрольных примеров показаны в приложении В.

1.8 Общие требования к программному продукту

Пользователи должны иметь базовые навыки пользования персональным компьютером.

Минимальные требования к техническому обеспечению программного продукта следующие:

* ОС:  Windows 7 Service Pack1/ Windows 7 64Bit Service Pack1/ Windows 8.1 64Bit / Windows 10 64Bit / Mac OS;
* процессор: 2.40 Ггц (четырехъядерный) / AMD Phenom 9850 (четырехъядерный) @ 2.5 Ггц;
* оперативная память: 512МБ (Win 7/Win8 и выше);
* видеокарта: NVIDIA 9800 GT c 512 МБ видеопамяти/ AMD HD 4870 с 1 Гб видеопамяти (DX 9, 10, 10.1);
* жесткий диск: 10 гигабайт свободного места;
* Microsoft DirectX версия 9.0c;

Функциональные возможности программного продукта:

* приложение должно формировать и отображать выходные данные пользователю;
* в приложении должен быть обеспечен просмотр таблиц (справочников) базы данных с возможность добавления, редактирования, удаления данных.

Требования к надежности:

* приложение должно обрабатывать ошибочные действия пользователя и сообщать ему об этом;
* приложение должно обеспечивать контроль входной и выходной информации.

Требования к информационной и программной совместимости: обеспечить работу приложения с таблицами СУБД MySQL.

2. Экспериментальный раздел

2.1 Описание программы

Программа имеет модульную структуру. При ее запуске выполняется проект на WPF MountaineeringClub.exe. Программа MountaineeringClub.exe написана на языке C# в среде программирования Visual Studio 2022 с использованием системы управления базой данных MS SQL Server 2022.

Схема взаимодействия модулей программы представлена на рисунке 2.1.1. 2.1.1. Описание модулей и методов представлено в таблице 2.1.1.

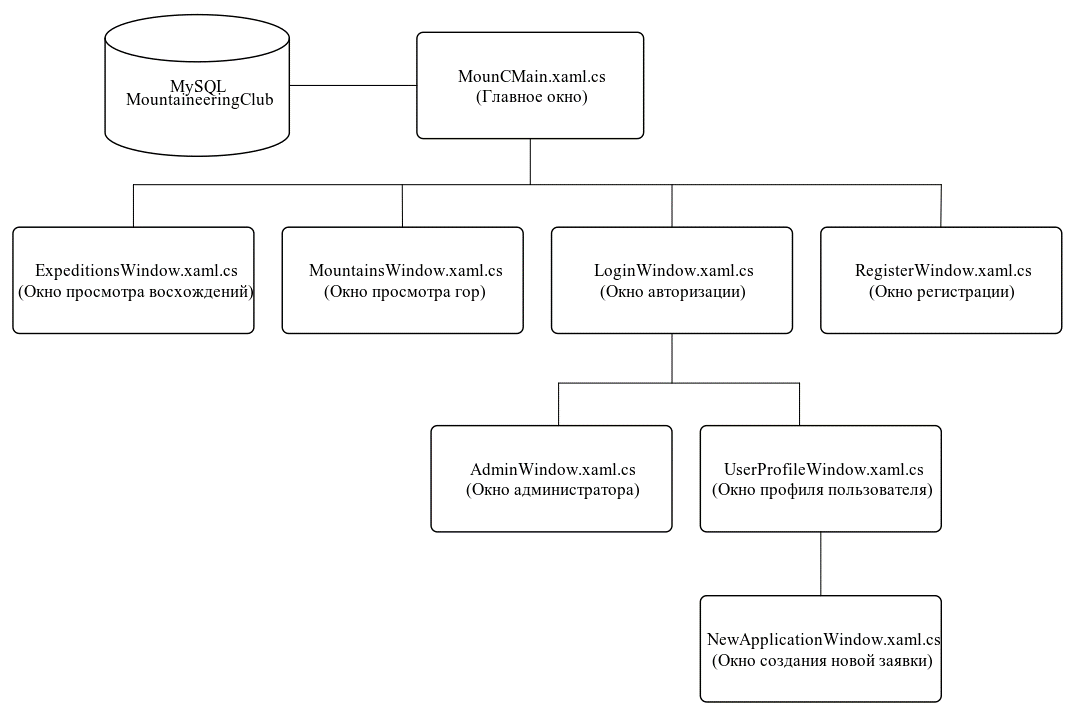


Рисунок 2.1.1 – Схема взаимодействия модулей

Таблица 2.1.1. - Описание модулей

|  |  |
| --- | --- |
| Методы | Назначение |
| 1 | 2 |
| MounCMain.xaml.cs | |
| public MounCMain() | Инициализация компонентов и вызов метода загрузки данных |
| private void LoadDatas() | Загрузка данных в список новостей и анонсов |
| private void LoginButton\_Click(object sender, RoutedEventArgs e) | Переход в окно авторизации |
| private void RegisterButton\_Click(object sender, RoutedEventArgs e) | Переход в окно регистрации |
| private void MountainsButton\_Click(object sender, RoutedEventArgs e) | Переход в окно с горами |
| private void ExpeditionsButton\_Click(object sender, RoutedEventArgs e) | Переход в окно с восхождениями |
| ExpeditionsWindow.xaml.cs | |
| public ExpeditionsWindow() | Инициализация компонентов и вызов метода загрузки данных |
| private void LoadAscents() | Загрузка данных в список восхождений |
| private void ApplyFiltersButton\_Click(object sender, RoutedEventArgs e) | Применение выбранных фильтров к списку |
| private void ResetFiltersButton\_Click(object sender, RoutedEventArgs e) | Сброс фильтров |
| private void BackButton\_Click(object sender, RoutedEventArgs e) | Возврат в главное окно |
| MountainsWindow.xaml.cs | |

Продолжение таблицы 2.1.1

|  |  |
| --- | --- |
| 1 | 2 |
| public MountainsWindow() | Инициализация компонентов и вызов метода загрузки данных |
| private void LoadMountains() | Загрузка данных в список гор |
| private void UpdateStatusText(int count) | Изменение сведений о кол-ве загруженных гор |
| private void ApplyFilters() | Применение выбранных фильтров к списку |
| private void BackButton\_Click(object sender, RoutedEventArgs e) | Возврат в главное окно |
| private void ResetFiltersButton\_Click(object sender, RoutedEventArgs e) | Сброс выбранных фильтров |
| private void RussianText\_PreviewTextInput(object sender, TextCompositionEventArgs e) | Валидация полей с разрешением на ввод букв русского алфавита |
| private void Number\_PreviewTextInput(object sender, TextCompositionEventArgs e) | Валидация полей с разрешением на ввод цифр |
| private void NameFilterTextBox\_TextChanged(object sender, TextChangedEventArgs e) | Вызов функции применения фильтров |
| private void CountryFilterTextBox\_TextChanged(object sender, TextChangedEventArgs e) | Вызов функции применения фильтров |
| private void HeightTextBox\_TextChanged(object sender, TextChangedEventArgs e) | Вызов функции применения фильтров |
| RegisterWindow.xaml.cs | |
| public RegisterWindow() | Инициализация компонентов |

Продолжение таблицы 2.1.1

|  |  |
| --- | --- |
| 1 | 2 |
| private void CancelButton\_Click(object sender, RoutedEventArgs e) | Переход в окно авторизации |
| private void RegisterButton\_Click(object sender, RoutedEventArgs e) | Вызов методов проверки полей для ввода и добавление пользователя в БД |
| private bool ValidateFields() | Проверка валидации введенных данных |
| private void ResetErrorMessages() | Сброс отображения неверных данных |
| private void ShowError(TextBlock errorTextBlock, string message) | Отображение предупреждающих об ошибках сообщений |
| private void ShowError(TextBox textBox, string message) | Отображение предупреждающих об ошибках полей |
| private bool IsValidEmail(string email) | Проверка валидности почты |
| private bool IsValidPhone(string phone) | Проверка валидности номера телефона |
| LoginWindow.xaml.cs | |
| public LoginWindow() | Инициализация компонентов |
| private void RegisterButton\_Click(object sender, RoutedEventArgs e) | Переход в окно регистрации |
| private void LoginButton\_Click(object sender, RoutedEventArgs e) | Проверка введенных данных и переход в окно пользователя/администратора |
| private void CancelButton\_Click(object sender, RoutedEventArgs e) | Переход в главное окно |
| UserProfileWindow.xaml.cs | |

Продолжение таблицы 2.1.1

|  |  |
| --- | --- |
| 1 | 2 |
| public UserProfileWindow(User user) | Инициализация компонентов и вызов метода загрузки данных |
| private void LoadDatas() | Загрузка данных в список заявок и восхождений |
| private int GetActiveApplicationsCount(int participantId) | Получение кол-ва актуальных заявок |
| private int GetActiveAscentsCount(int participantId) | Получение кол-ва выполненных восхождений |
| private void LogoutButton\_Click(object sender, RoutedEventArgs e) | Переход в главное окно |
| private void NewApplicationButton\_Click(object sender, RoutedEventArgs e) | Открытие окна создания новой заявки |
| private void CancelButton\_Click(object sender, RoutedEventArgs e) | Вызов метода переключения данных о пользователе в режим просмотра |
| private void EditProfileButton\_Click(object sender, RoutedEventArgs e) | Переключение данных о пользователе в режим редактирования |
| private void SaveButton\_Click(object sender, RoutedEventArgs e) | Проверка валидации полей и сохранение обновленных данных о пользователе |
| private void SwitchToViewMode() | Переключение данных о пользователе в режим просмотра |

Продолжение таблицы 2.1.1

|  |  |
| --- | --- |
| 1 | 2 |
| private void LeaveTheClubButton\_Click(object sender, RoutedEventArgs e) | Изменение статуса пользователя и переход в главное окно |
| NewApplicationWindow.xaml.cs | |
| public NewApplicationWindow(User user) | Инициализация компонентов и вызов метода загрузки данных |
| private void LoadPlannedAscents() | Загрузка запланированных восхождений |
| private void CancelButton\_Click(object sender, RoutedEventArgs e) | Возврат в окно пользователя |
| private void SubmitButton\_Click(object sender, RoutedEventArgs e) | Проверка введенных данных и подача заявки |
| private void MountainComboBox\_SelectionChanged(object sender, System.Windows.Controls.SelectionChangedEventArgs e) | Загрузка даты и описания выбранного восхождения |

Код программы представлен в приложении Г.

2.2 Протокол тестирования программного продукта

В ходе тестирования программного продукта на корректных и некорректных данных не было обнаружено ошибок, которые влияли бы на работу самого программного продукта и всей системы.

Данный программный продукт удовлетворяет всем предъявленным требованиям, имеет комфортный интерфейс и интуитивно понятный функционал, исключает появления системных ошибок.

Таблица 2.2.1 – Тестирование авторизации пользователя с некорректными данными

|  |  |
| --- | --- |
| Поле | Описание |
| 1 | 2 |
| Дата теста | 13.06.2025 |
| Приоритет тестирования (*Низкий/Средний/Высокий*) | Высокий |
| Заголовок/название теста | Тестирование авторизации с некорректными данными |
| Резюме испытания | Проверка отказа в авторизации при вводе некорректных данных |
| Этапы теста | 1 Запуск приложения;  2 Нажатие на кнопку «Вход»;  3 Ввод номера телефона или почты;  4 Ввод пароля;  5 Нажатие на кнопку «Войти». |
| Тестовые данные | 1 Номер телефона «89378415022»;  2 пароль «qweqwe». |
| Ожидаемый результат | Отображение сообщения «Неверный логин или пароль» |
| Фактический результат | Сообщение отобразилось, вход в аккаунт не произошел |

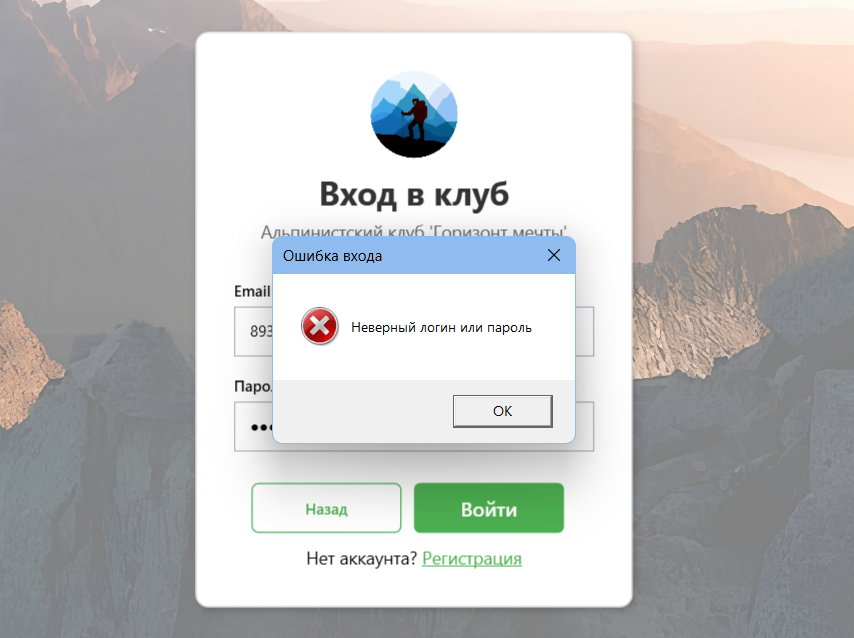


Рисунок 2.2.1 – Результат авторизации с некорректными данными

Таблица 2.2.2 – Тестирование авторизации пользователя с корректными данными

|  |  |
| --- | --- |
| Поле | Описание |
| 1 | 2 |
| Дата теста | 13.06.2025 |
| Приоритет тестирования (*Низкий/Средний/Высокий*) | Высокий |
| Заголовок/название теста | Тестирование авторизации пользователя с корректными данными |
| Резюме испытания | Проверка успешной авторизации при вводе корректных данных |
| Этапы теста | 1 Запуск приложения;  2 Нажатие на кнопку «Вход»;  3 Ввод номера телефона или почты;  4 Ввод пароля;  5 Нажатие на кнопку «Войти». |
| Тестовые данные | 1 Номер телефона «89378415022»;  2 пароль «qwe123». |
| Ожидаемый результат | Переход в окно пользователя. |
| Фактический результат | Переход в окно пользователя прошел успешно. |

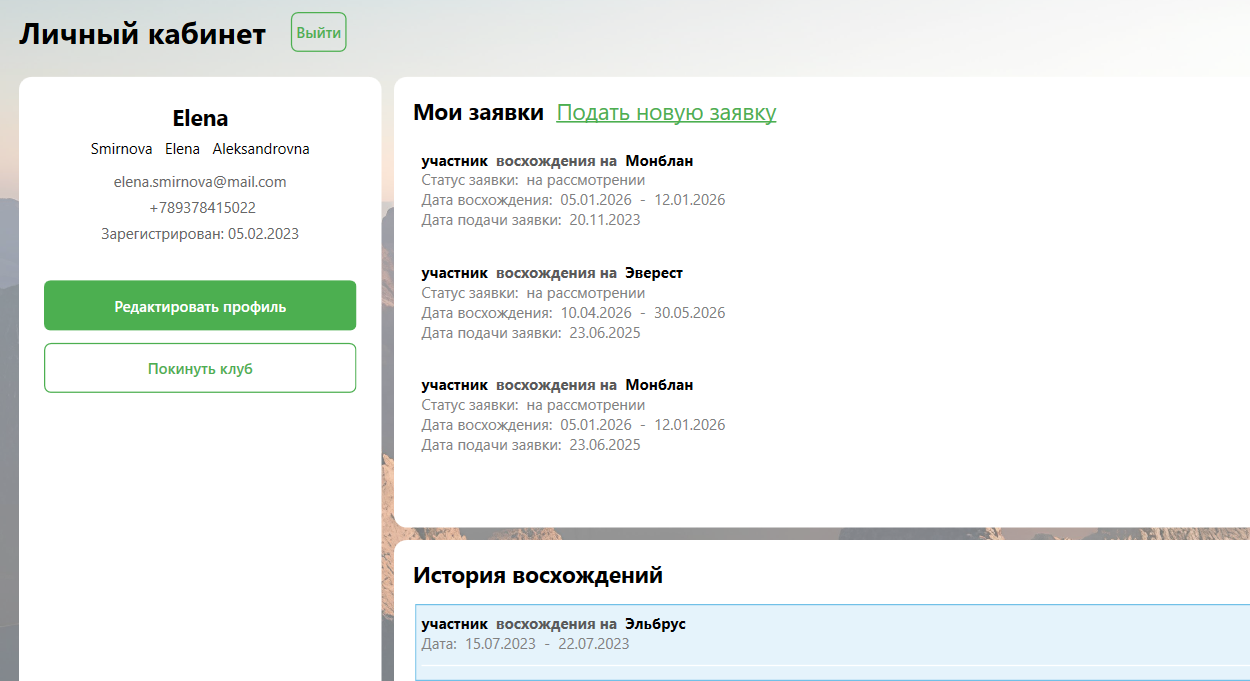


Рисунок 2.2.2 – Результат тестирования авторизации пользователя с корректными данными

В протоколе тестирования отражаются:

− тестирование на корректных данных;

− тестирование на некорректных данных;

− тестировании продукта на данных контрольного примера.

Тестирование на создание новой заявки при корректных данных, ожидаемое сообщение «Заявка успешно подана!» (рисунок 2.2.3).

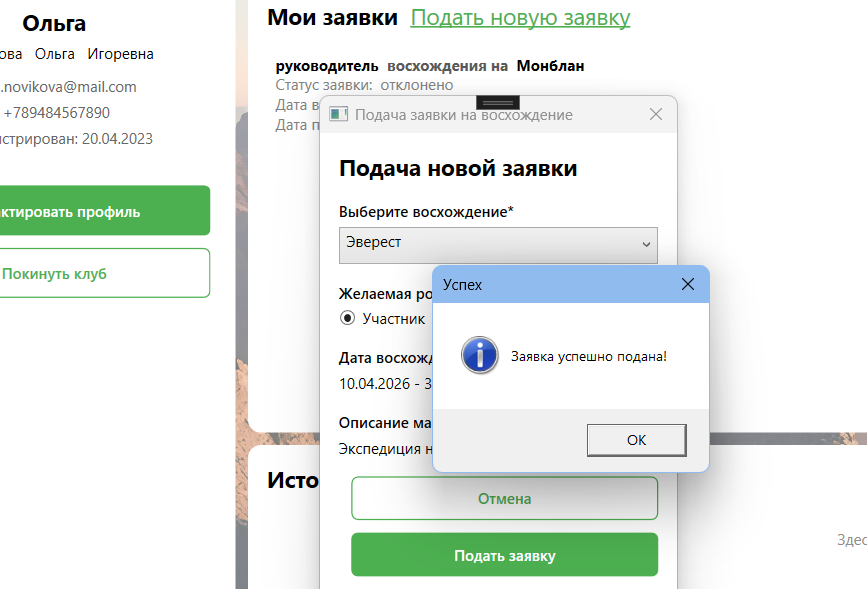


Рисунок 2.2.3 – Сообщение об успешной подаче заявки на восхождение

Тестирование на редактирование данных пользователя при корректных данных, ожидаемое сообщение «Данные успешно сохранены!» (рисунок 2.2.4).

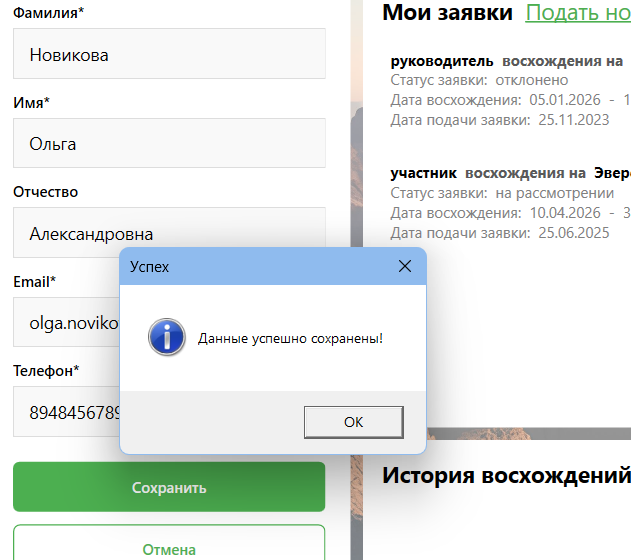


Рисунок 2.2.4 – Сообщение об успешном редактировании данных пользователя

Тестирование на подачу заявки на восхождение, на которое ранее была подана заявка, ожидаемое сообщение «У вас уже есть заявки на пересекающиеся восхождения» (рисунок 2.2.5).

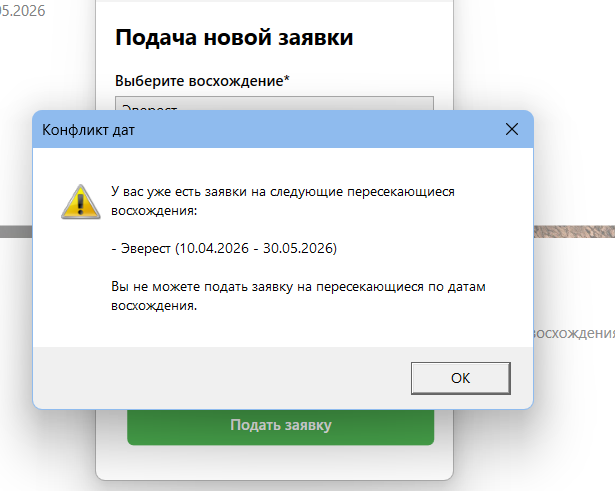


Рисунок 2.2.5 – Сообщение о безуспешной подаче заявки на восхождение

2.3 Руководство пользователя

Назначение системы

Программа «Организация деятельности альпинистского клуба» предназначена для автоматизации работы альпинистского клуба, включая планирование восхождений, обработку заявок участников, ведение базы данных гор и маршрутов, а также формирование отчетности.

Основной целью данной информационной системы является обеспечить автоматизацию процесса подготовки, печати и хранения банковских платежных поручений. База данных должна предоставлять возможность автоматического заполнения полей.

Программное обеспечение разрабатывается для персональной вычислительной техники со следующими характеристиками:

* Microsoft Windows 7 / 8 / 10 / 11;
* процессор 1 ГГц и выше;
* 512 МБ ОЗУ;
* 100 МБ свободного пространства на диске;
* разрешение экрана монитора не менее 1280 × 720;
* дополнительное оборудование: принтер.

Программа «Организация деятельности альпинистского клуба» рассчитана на пользователей с базовыми навыками работы в графическом интерфейсе Windows. Для удобства реализован интуитивно понятный интерфейс с подсказками.

Подготовка системы к работе

Для запуска программы необходимо запустить приложение MountaineeringClub.exe из каталога, в котором установлен данный программный продукт. После этого открывается главное окно клуба (рисунок 2.3.1).

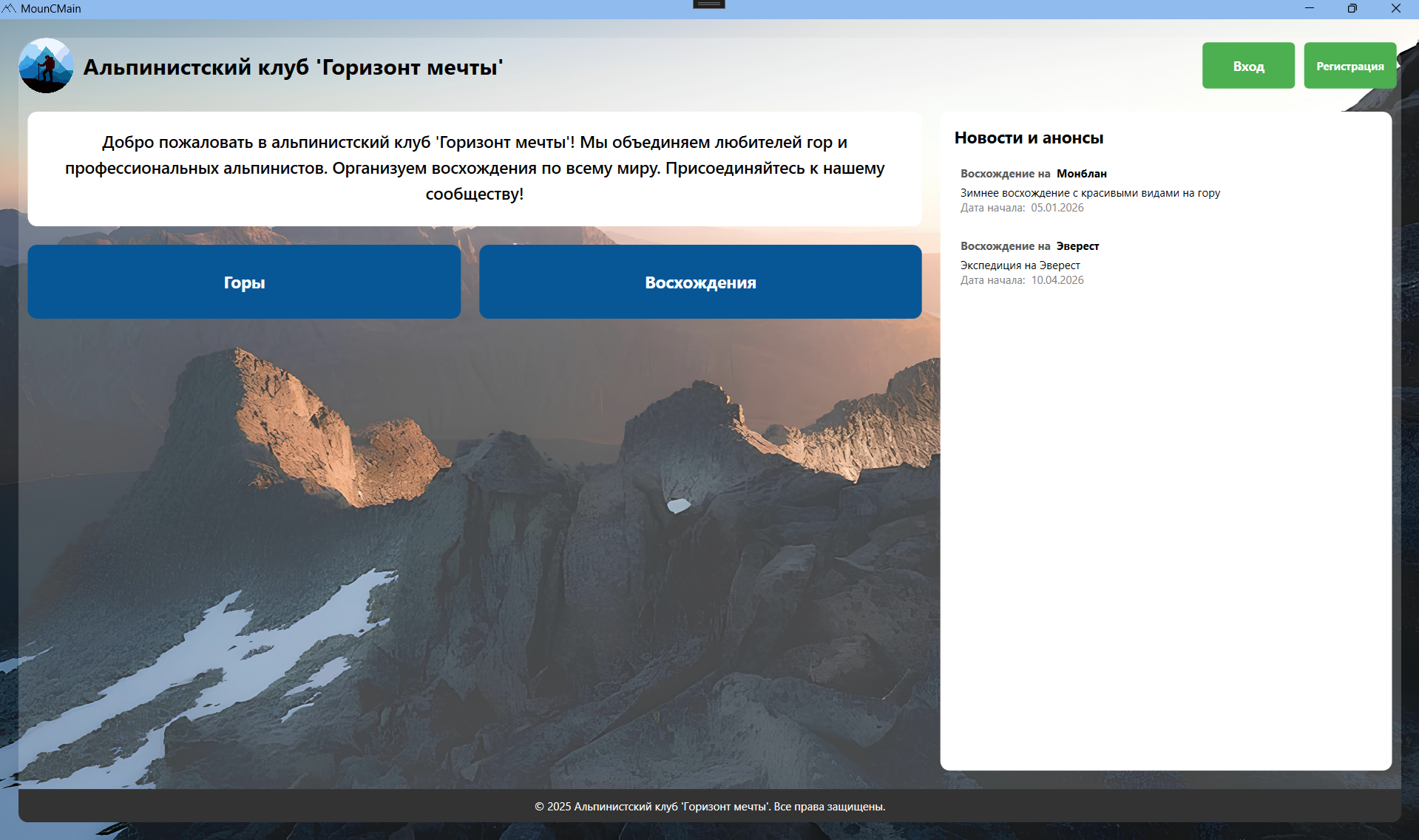


Рисунок 2.3.1 – Главное окно клуба

Описание операций

При нажатии на кнопки «Вход», «Регистрация», «Горы», «Восхождения» произойдет переход в соответствующие окна.

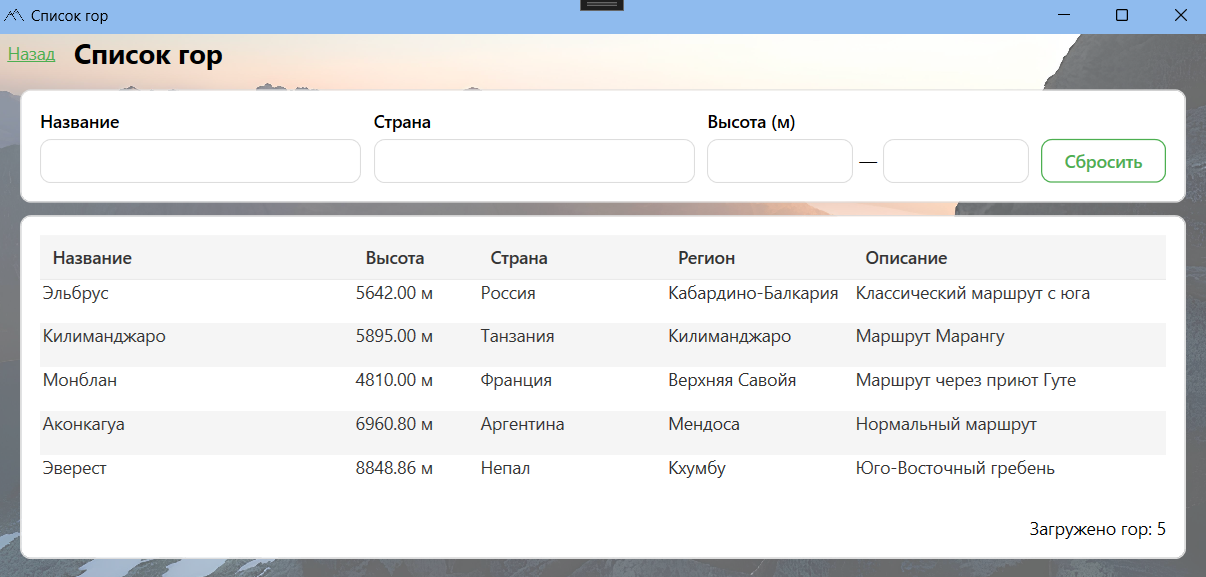


Рисунок 2.3.2 – Интерфейс приложения, в случае нажатия кнопки «Горы»

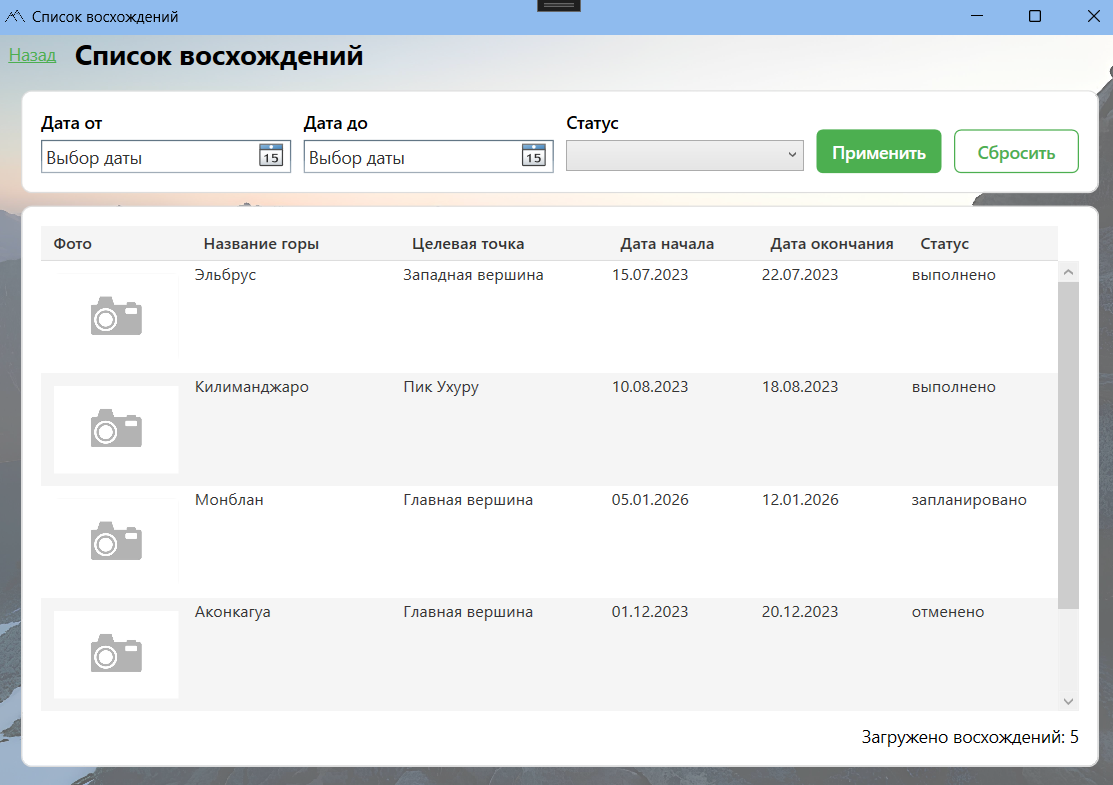


Рисунок 2.3.3 – Интерфейс приложения, в случае нажатия кнопки «Восхождения»

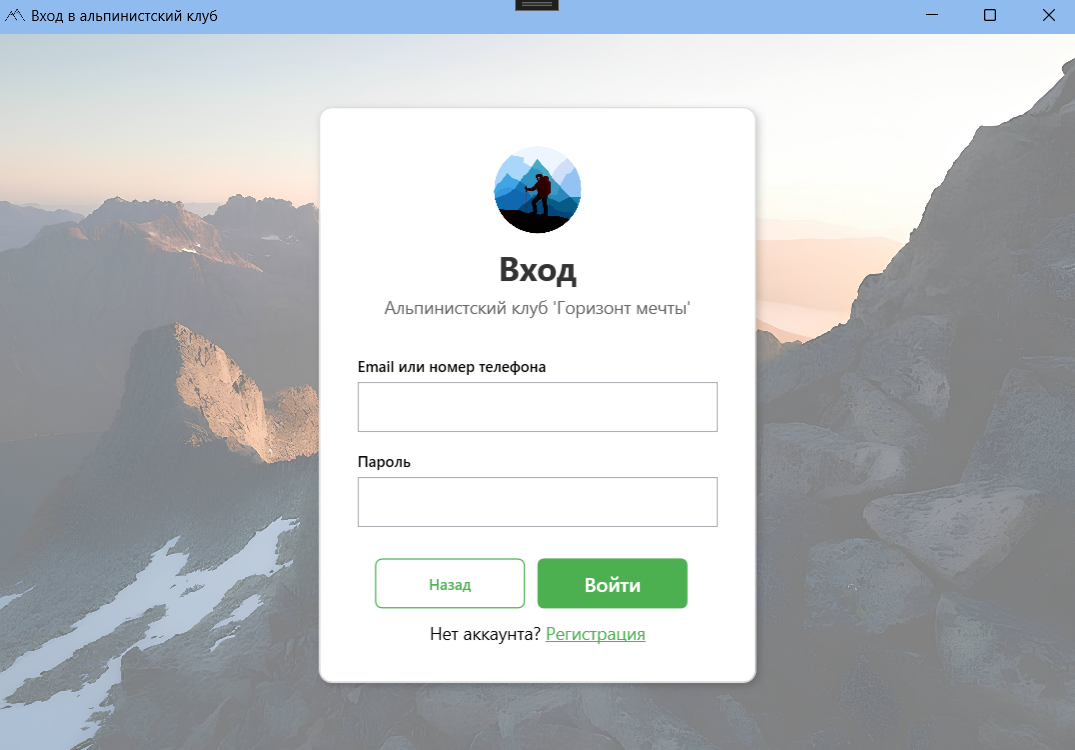


Рисунок 2.3.4 – Интерфейс приложения, в случае нажатия кнопки «Вход»

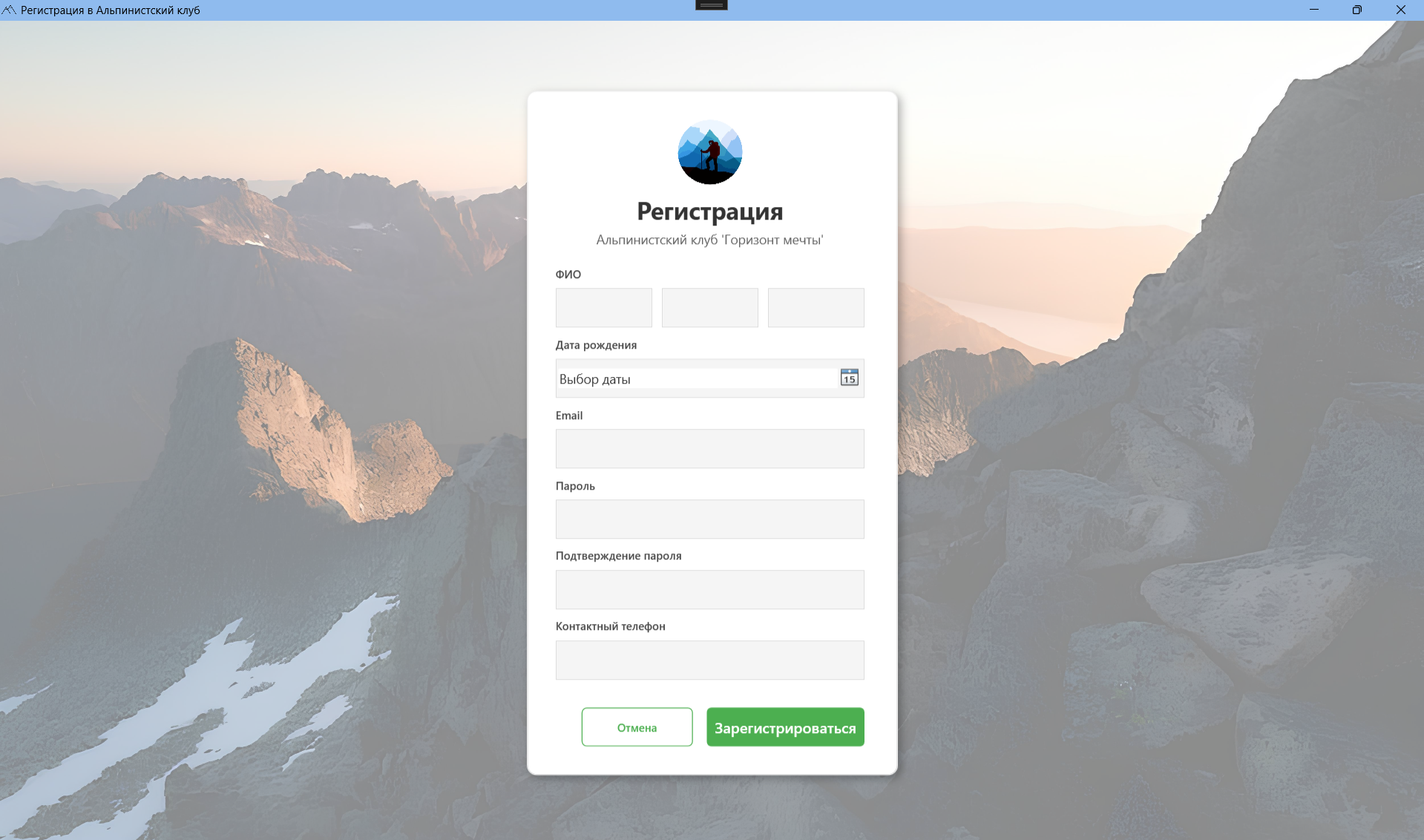


Рисунок 2.3.5 – Интерфейс приложения, в случае нажатия кнопки «Регистрация»

В окне «Список гор» написав в соответствующие поля для ввода название/страну/минимальную высоту/максимальную высоту, и нажав кнопку «Применить», высветятся только те горы, что соответствуют введенным вами критериям.

Для отображения изначального списка необходимо нажать на кнопку «Сбросить».

Для возврата на главное окно необходимо нажать на кнопку «Назад» в левом верхнем углу.

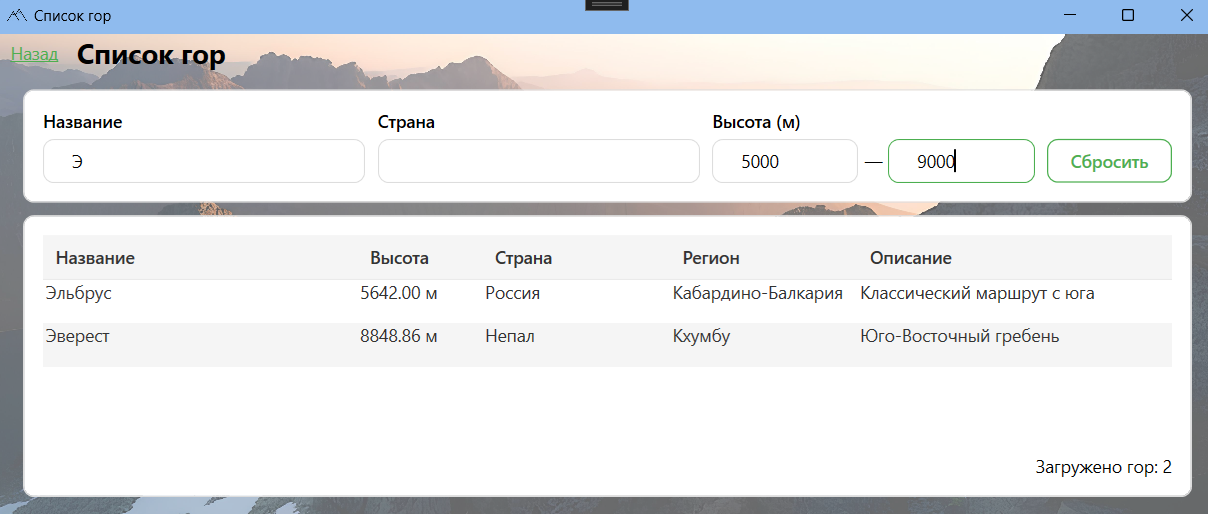


Рисунок 2.3.6 – Отображение списка гор в соответствии с введенными критериями

В окне «Список восхождений» выбрав в соответствующих полях начальную дату, конечную дату, статус восхождения и нажав кнопку «Применить», высветятся только те восхождения, что соответствуют выбранным вами критериям.

Для отображения изначального списка необходимо нажать на кнопку «Сбросить».

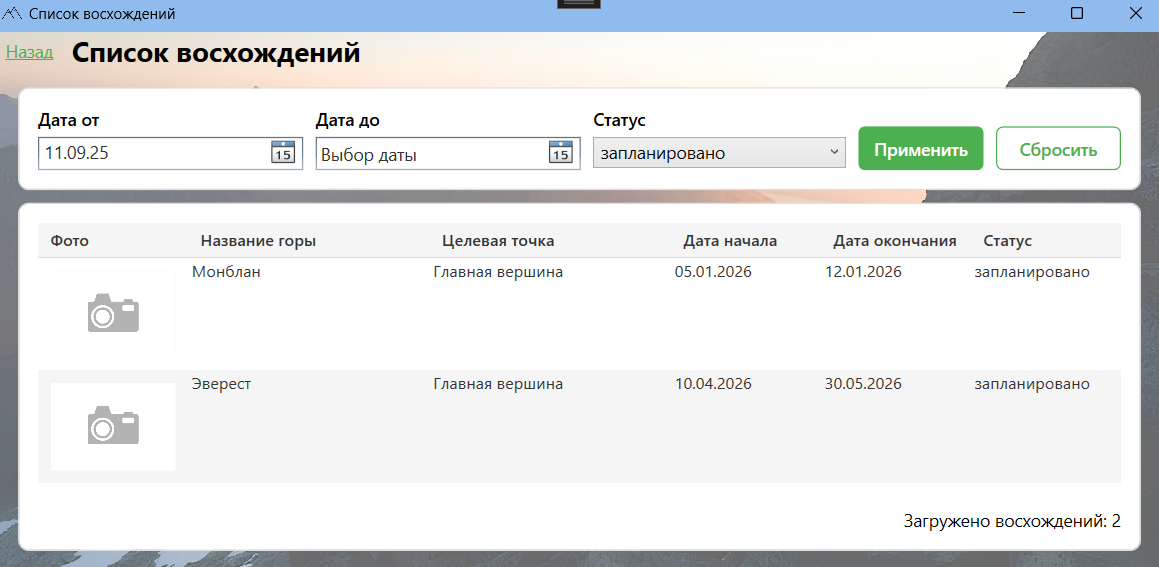


Рисунок 2.3.7 – Отображение списка восхождений в соответствии с введенными критериями

Для возврата на главное окно необходимо нажать на кнопку «Назад» в левом верхнем углу.

Для осуществления регистрации пользователя в окне «Регистрация» необходимо ввести данные в поля для ввода и нажать на кнопку «Зарегистрироваться».

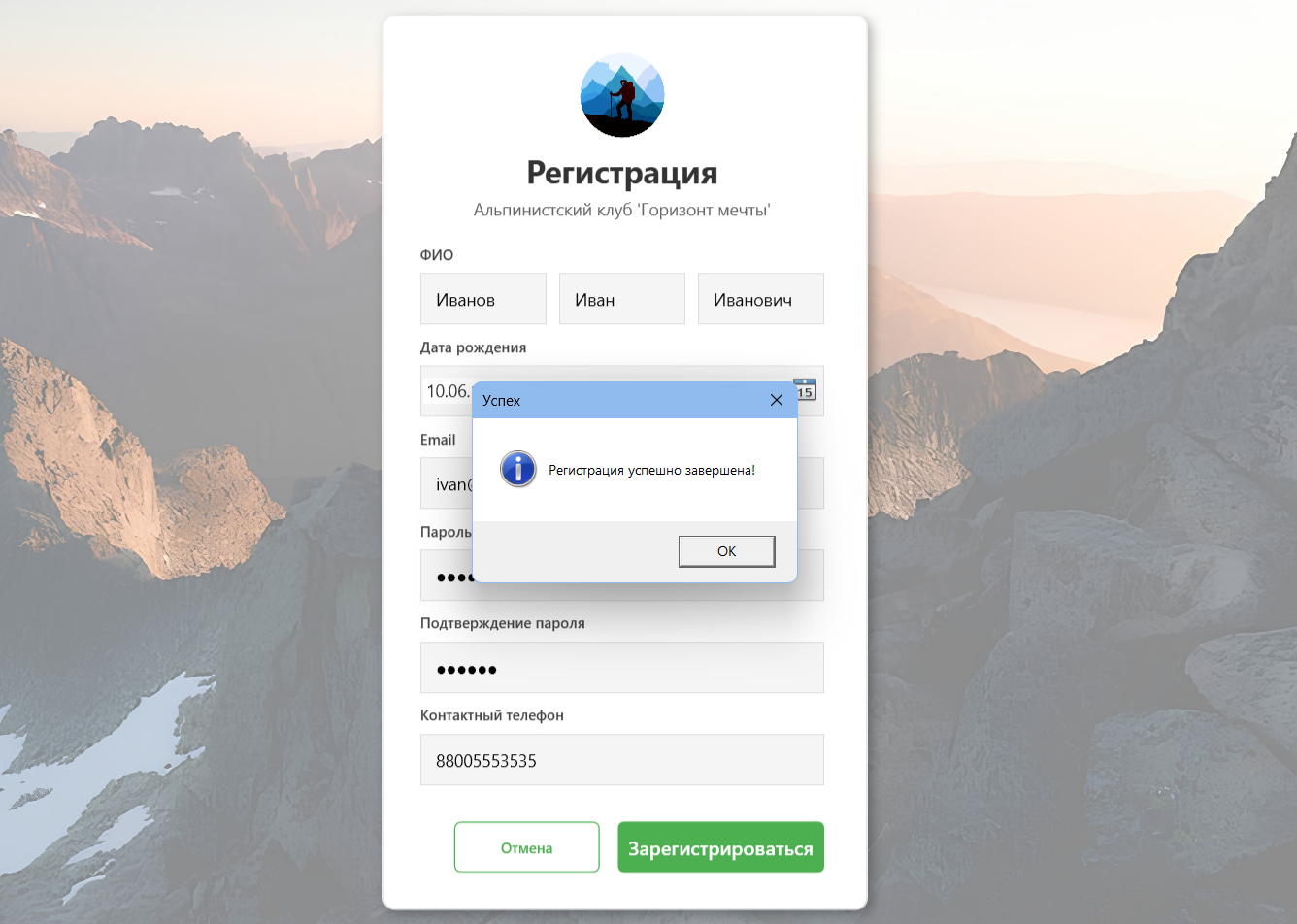


Рисунок 2.3.8 – Регистрация пользователя

Для осуществления регистрации пользователя в окне «Вход» необходимо ввести данные в поля для ввода и нажать на кнопку «Войти».

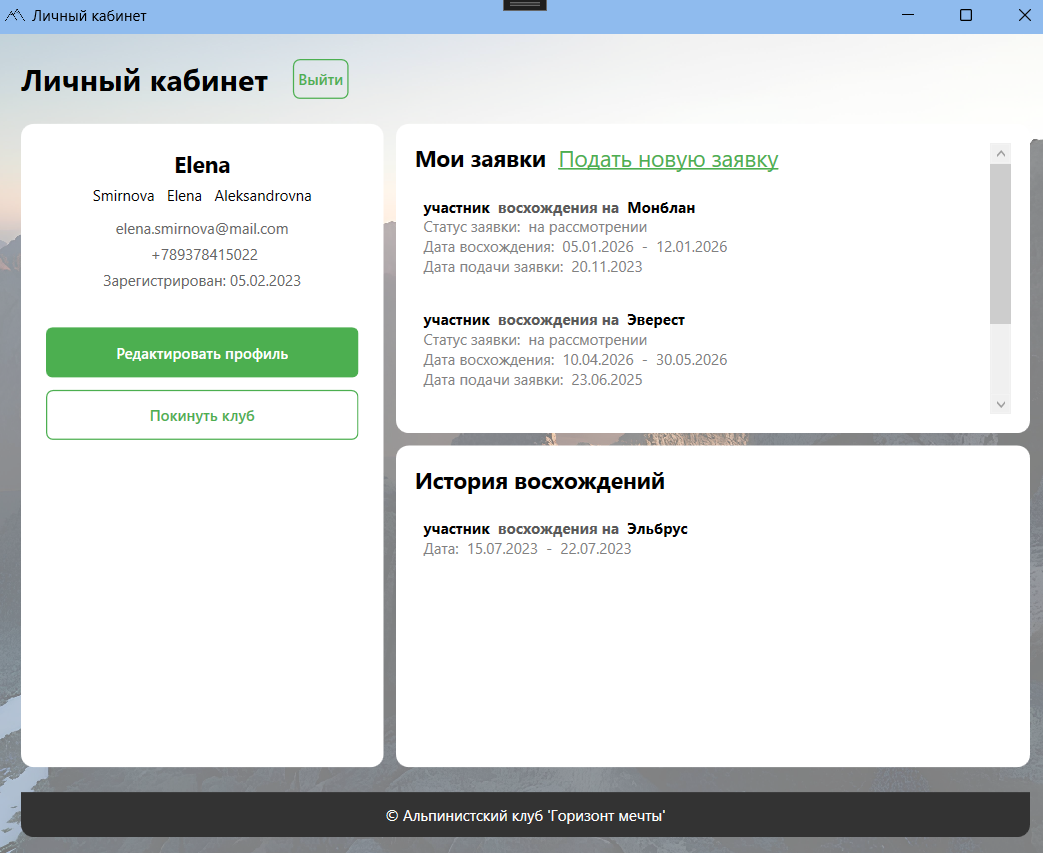


Рисунок 2.3.9 – Интерфейс пользователя в случае ввода данных «зарегистрированного участника» в окне «Вход»

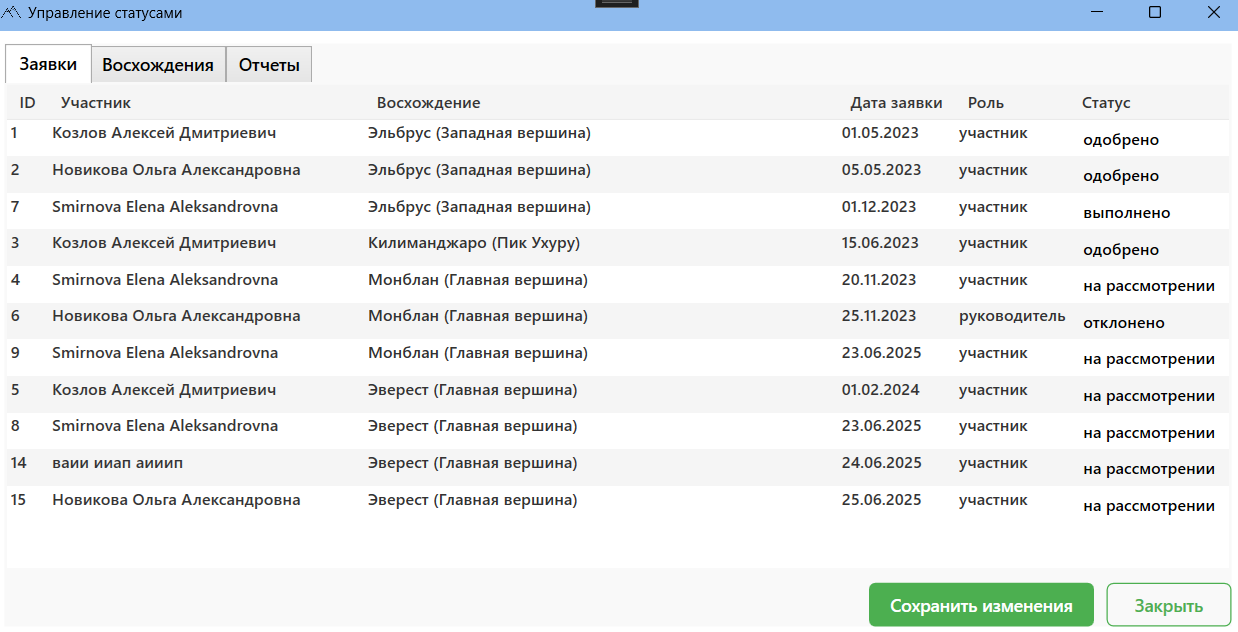


Рисунок 2.3.9 – Интерфейс пользователя в случае ввода данных «администратора» в окне «Вход»

В окне «Личный кабинет» при нажатии на кнопку «Редактировать профиль», можно изменить свои данные и сохранить изменения нажатием на кнопку «Сохранить».

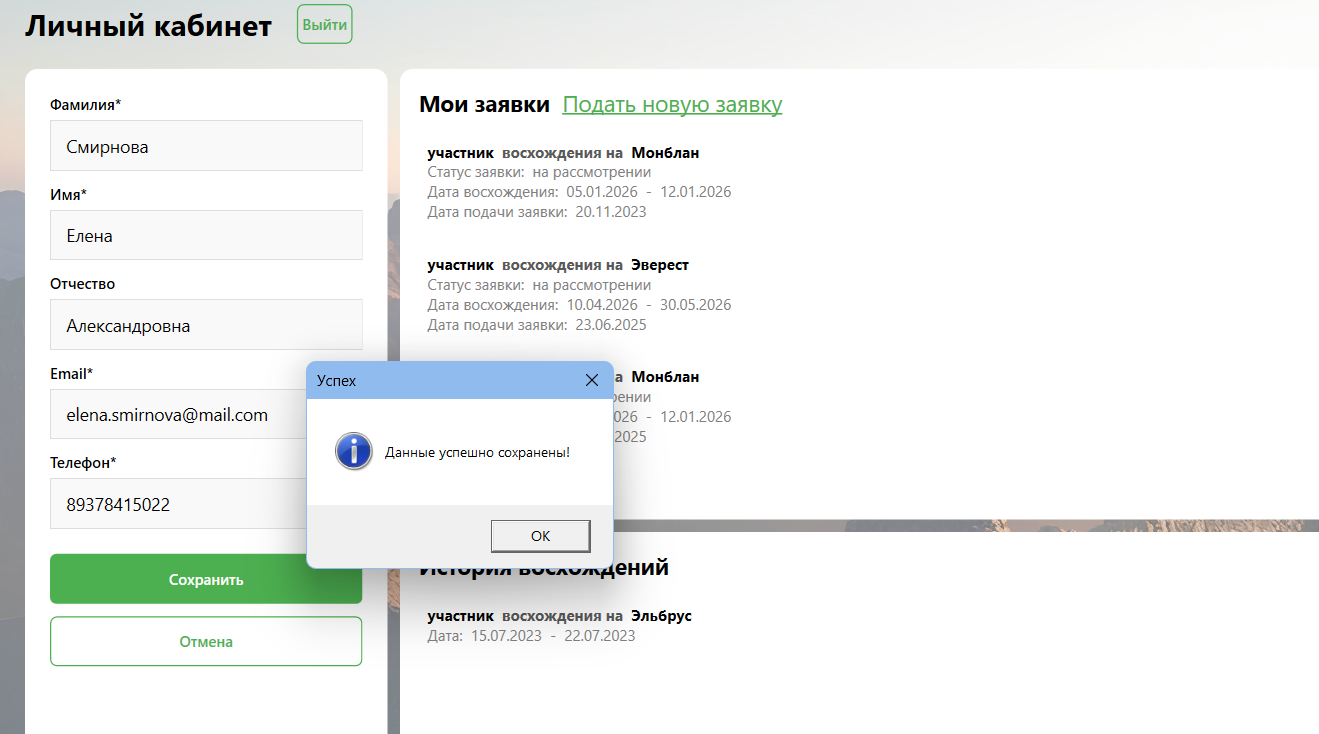


Рисунок 2.3.10 – Редактирование данных пользователя в окне «Личный кабинет»

Для подачи заявки на восхождение необходимо нажать на кнопку «Подать новую заявку», сделать свой выбор в пользу какого-либо восхождения, выбрать роль и нажать на кнопку «Подать заявку».

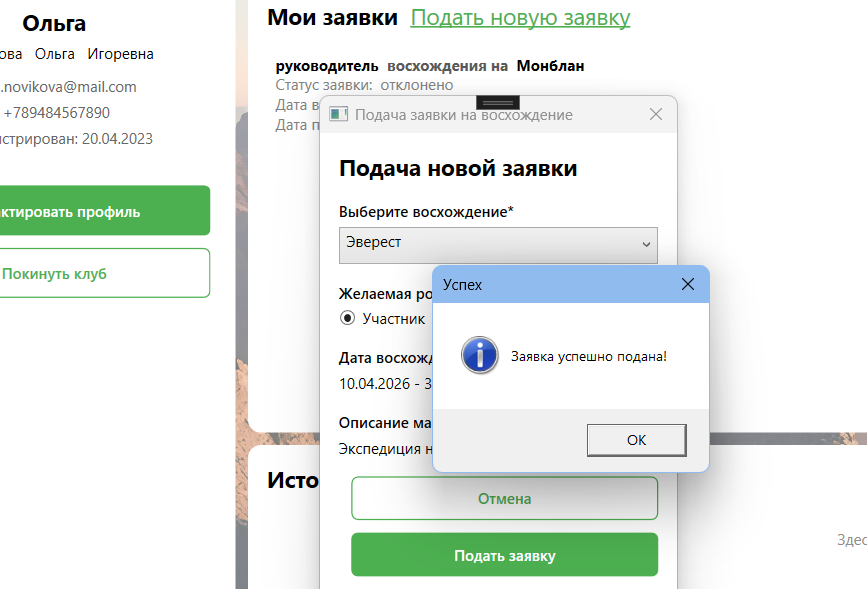


Рисунок 2.3.11 – Подача заявки на восхождение в окне «Личный кабинет»

Для изменения статуса заявок или статуса восхождений в окне администратора необходимо 2 раза щелкнуть в столбце «Статус» на строчку, которую вы хотите отредактировать и после этого нажать на кнопку «Сохранить изменения».

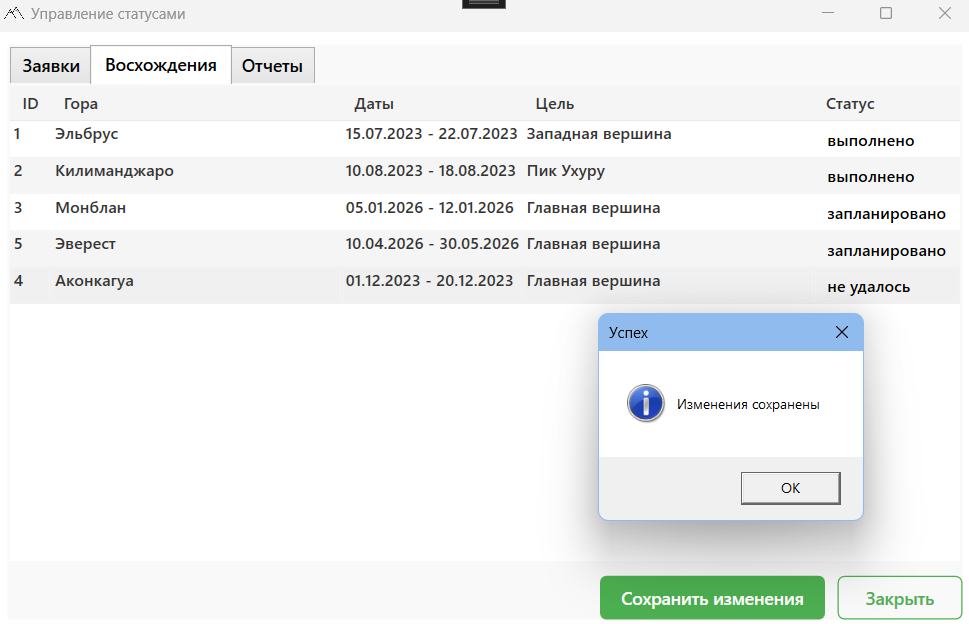


Рисунок 2.3.12 – Изменение статуса восхождения в окне «Окно администратора»

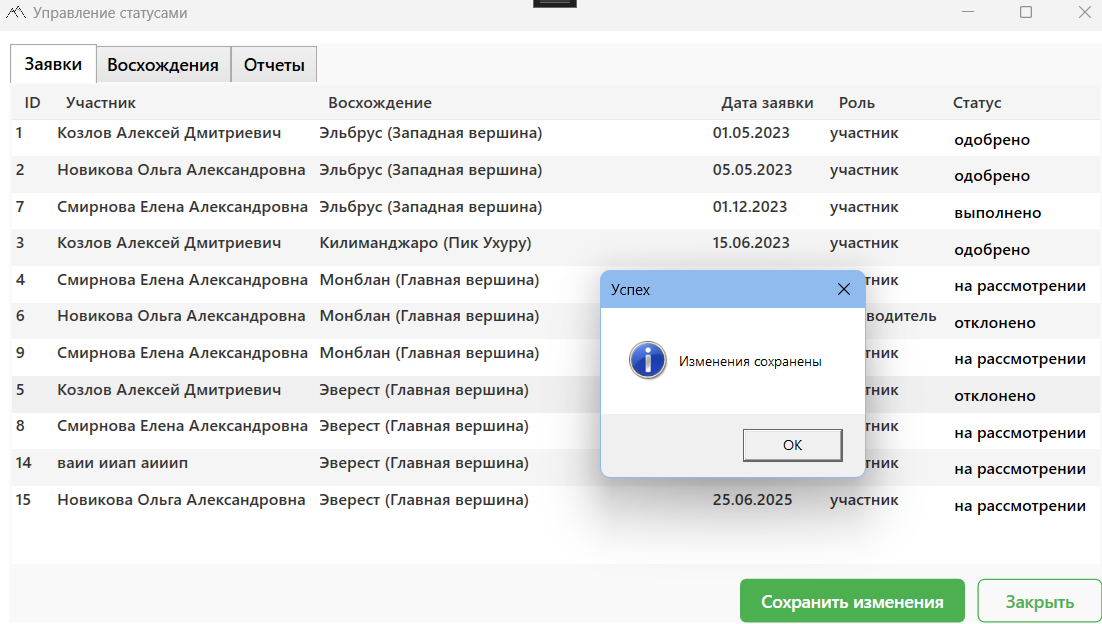


Рисунок 2.3.13 – Изменение статуса заявки в окне «Окно администратора»

Для формирования отчета в окне администратора необходимо выбрать вид отчета, который вам необходим и нажать на кнопку «Сформировать».

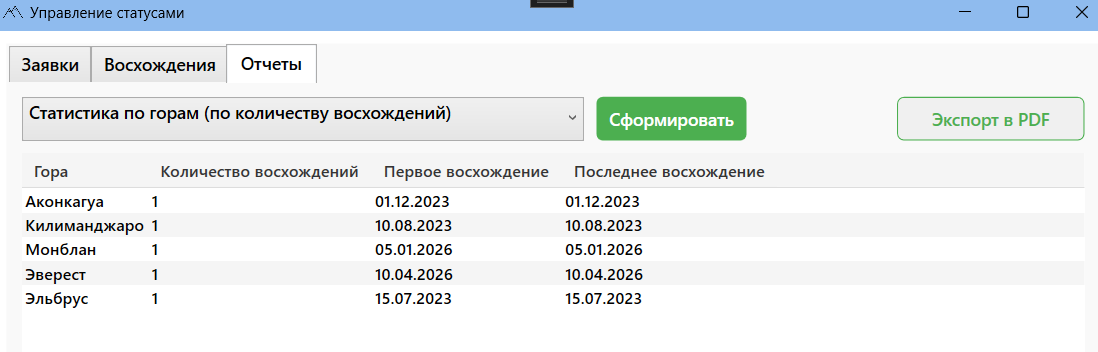


Рисунок 2.3.14 – Отображение сформированного отчета в окне администратора

Для экспортирования отчета в PDF необходимо нажать на кнопку «Экспорт в PDF».

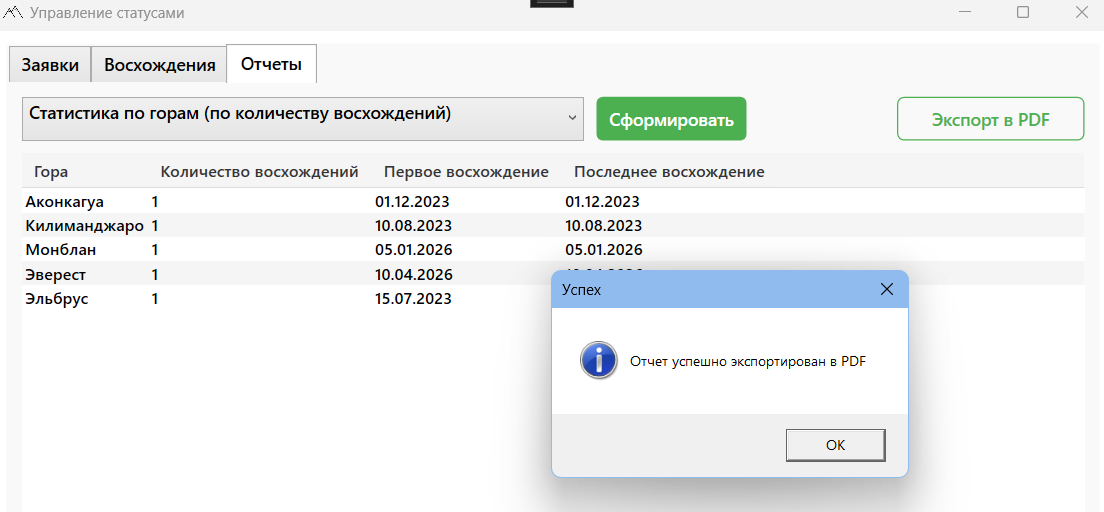


Рисунок 2.3.15 – Экспортирование отчета в PDF в окне администратора



Рисунок 2.3.16 – Сформированный PDF отчет о горах

ЗАКЛЮЧЕНИЕ

В ходе выполнения курсового проекта была разработана архитектура и реализована логика работы WPF-приложения «Организация деятельности альпинистского клуба».

При этом были изучены особенности реализации компонентов WPF для построения клиентских приложений с визуально привлекательными возможностями взаимодействия с пользователем.

Результатом работы стало создание WPF-приложения для сокращения времени администраторов альпинистского клуба на обработку заявок и планирование восхождений, автоматизации формирования отчетов и статистики.

WPF-приложение написано на языке C# в среде разработки Visual Studio 2022 с использованием языка разметки XAML и системы управления базой данных My SQL Workbench 2022.

Были проведены опытная эксплуатация и отладочное тестирование WPF приложения. По результатам отладочного тестирования были устранены некоторые недостатки, в частности были выявлены и исправлены следующие недостатки: улучшена обработка ввода, оптимизирован экспорт данных в печатные формы. Дополнительно было подготовлено руководство пользователя, включающее инструкции для администраторов и участников клуба.

С помощью приложения на основании данных контрольного примера были получены результаты, которые полностью совпадают с выходной информацией контрольного примера

Входные данные контрольного примера

Таблица А.1 – Пользователи системы

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ID пользователя | Имя пользователя | Фамилия | Отчество | Номер телефона | Электронная почта | Пароль | Дата рождения | Дата регистрации | Роль |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 | Иван | Петров | Сергеевич | 89161234567 | ivan.petrov@mail.com | admin123 | 1985-03-15 | 2023-01-10 | администратор |
| 2 | Елена | Смирнова | Александровна | 89378415022 | elena.smirnova@mail.com | qwe123 | 1990-07-22 | 2023-02-05 | зарегистрированный участник |
| 3 | Алексей | Козлов | Дмитриевич | 89373456789 | alex.kozlov@mail.com | йцу123 | 1988-11-30 | 2023-03-12 | зарегистрированный участник |
| 4 | Ольга | Новикова | Александровна | 89484567890 | olga.novikova@mail.com | hashed\_password\_4 | 1992-05-18 | 2023-04-20 | зарегистрированный участник |
| 5 | Дмитрий | Васильев | Олегович | 89595678901 | dmitry.vasiliev@mail.com | hashed\_password\_5 | 1983-09-14 | 2023-01-25 | администратор |
| 6 | Сергей | Иванов | Анатольевич | 89696789012 | sergey.ivanov@mail.com | hashed\_password\_6 | 1979-12-05 | 2023-02-18 | администратор |
| 7 | ииап | ваии | аииип | 89999999999 | маав@hbail.vion | qwe123 | 2011-06-05 | 2025-06-24 | зарегистрированный участник |
| 8 | Иван | Иванов | Иванович | 88005553535 | ivan@gmail.com | Qwe123 | 2011-06-10 | 2025-06-25 | зарегистрированный участник |
| 9 | Анна | Соколова | Владимировна | 89160234567 | anna.sokolova@mail.com | securepass1 | 1991-04-12 | 2023-05-15 | зарегистрированный участник |
| 10 | Михаил | Лебедев | Игоревич | 89262345678 | m.lebedev@mail.com | securepass2 | 1987-08-24 | 2023-06-20 | зарегистрированный участник |
| 11 | Татьяна | Кузнецова | Сергеевна | 89373056789 | t.kuznetsova@mail.com | securepass3 | 1993-01-30 | 2023-07-10 | зарегистрированный участник |
| 12 | Артем | Попов | Александрович | 89480567890 | artem.popov@mail.com | securepass4 | 1989-11-05 | 2023-08-05 | зарегистрированный участник |

Продолжение таблицы А.1

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 13 | Екатерина | Васильева | Дмитриевна | 89505678901 | e.vasileva@mail.com | securepass5 | 1990-03-18 | 2023-09-12 | зарегистрированный участник |
| 14 | Павел | Семенов | Олегович | 89690789012 | p.semenov@mail.com | securepass6 | 1985-07-22 | 2023-10-20 | зарегистрированный участник |
| 15 | Мария | Голубева | Алексеевна | 89797890123 | m.golubeva@mail.com | securepass7 | 1992-12-08 | 2024-01-15 | зарегистрированный участник |
| 16 | Денис | Виноградов | Викторович | 89898901234 | d.vinogradov@mail.com | securepass8 | 1988-05-25 | 2024-02-10 | зарегистрированный участник |
| 17 | Наталья | Белова | Игоревна | 89999012345 | n.belova@mail.com | securepass9 | 1994-09-14 | 2024-03-05 | зарегистрированный участник |
| 18 | Андрей | Комаров | Сергеевич | 89000123456 | a.komarov@mail.com | securepass10 | 1986-02-28 | 2024-04-18 | зарегистрированный участник |
| 19 | Светлана | Орлова | Андреевна | 89111234567 | s.orlova@mail.com | securepass11 | 1995-06-17 | 2024-05-22 | зарегистрированный участник |
| 20 | Константин | Андреев | Дмитриевич | 89222345678 | k.andreev@mail.com | securepass12 | 1984-10-09 | 2024-06-30 | зарегистрированный участник |

Таблица А.2 – Фотографии

|  |  |  |  |
| --- | --- | --- | --- |
| ID фотографии | ID восхождения | Путь | Описание изображения |
| 1 | 2 | 3 | 4 |
| 1 | 1 | BLOB | Вид с приюта Бочки |
| 2 | 1 | BLOB | Группа на вершине Эльбруса |
| 3 | 2 | BLOB | Рассвет на Килиманджаро |
| 4 | 2 | BLOB | Группа у знака Ухуру |
| 5 | 3 | BLOB | Подготовка к зимнему восхождению |
| 6 | 6 | BLOB | Вид с базового лагеря Макалу |
| 7 | 6 | BLOB | Группа на вершине Макалу |

Продолжение таблицы А.2

|  |  |  |  |
| --- | --- | --- | --- |
| 8 | 7 | BLOB... | Подготовка к восхождению на Чогори |
| 9 | 9 | BLOB | Панорама с Дхаулагири |
| 10 | 9 | BLOB | Команда в базовом лагере |
| 11 | 12 | BLOB | Рассвет на Аннапурне |
| 12 | 12 | BLOB | Группа на вершине |
| 13 | 14 | BLOB | Вид с Броуд-Пика |
| 14 | 15 | BLOB | Зимний пейзаж Гашербрума II |
| 15 | 15 | BLOB | Команда после успешного восхождения |

Выходные данные контрольного примера

Таблица Б.1 – Участники

|  |  |  |  |
| --- | --- | --- | --- |
| ID участника | ID пользователя | Статус | История восхождений |
| 1 | 3 | активный | Эльбрус 2022, Килиманджаро 2023 |
| 2 | 4 | активный | Эльбрус 2023 |
| 3 | 2 | активный |  |
| 5 | 7 | активный |  |
| 6 | 8 | активный |  |
| 7 | 9 | активный | Эльбрус 2023, Монблан 2024 |
| 8 | 10 | активный | Килиманджаро 2024 |
| 9 | 11 | активный | Эльбрус 2024 |
| 10 | 12 | активный | Аконкагуа 2024 |
| 11 | 13 | активный | Эльбрус 2023, Килиманджаро 2024 |
| 12 | 14 | активный | Монблан 2023 |
| 13 | 15 | активный | Эльбрус 2024 |
| 14 | 16 | активный | Килиманджаро 2023, Аконкагуа 2024 |
| 15 | 17 | активный | Эльбрус 2024 |
| 16 | 18 | активный | Монблан 2024 |
| 17 | 19 | активный | Килиманджаро 2024 |
| 18 | 20 | активный | Эльбрус 2023, Монблан 2024 |
| 1 | 3 | активный | Эльбрус 2022, Килиманджаро 2023 |

Таблица Б.2 – Горы

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID горы | Название | Высота | Страна | Регион | Маршрут |
| 1 | Эльбрус | 5642.00 | Россия | Кабардино-Балкария | Классический маршрут с юга |
| 2 | Килиманджаро | 5895.00 | Танзания | Килиманджаро | Маршрут Марангу |
| 3 | Монблан | 4810.00 | Франция | Верхняя Савойя | Маршрут через приют Гуте |
| 4 | Аконкагуа | 6960.80 | Аргентина | Мендоса | Нормальный маршрут |
| 5 | Эверест | 8848.86 | Непал | Кхумбу | Юго-Восточный гребень |
| 6 | Макалу | 8485.00 | Непал | Кхумбу | Стандартный маршрут с севера |
| 7 | Чогори | 8611.00 | Пакистан | Балторо | Классический маршрут Абруцци |
| 8 | Лхоцзе | 8516.00 | Непал | Кхумбу | Южная стена |
| 9 | Дхаулагири | 8167.00 | Непал | Дхаулагири | Северо-восточный гребень |
| 10 | Манаслу | 8163.00 | Непал | Манаслу | Северо-восточный склон |
| 11 | Нанга Парбат | 8126.00 | Пакистан | Гилгит-Балтистан | Маршрут Диамира |
| 12 | Аннапурна | 8091.00 | Непал | Гандаки | Северная стена |
| 13 | Гашербрум I | 8080.00 | Пакистан | Балторо | Юго-западный склон |
| 14 | Броуд-Пик | 8051.00 | Пакистан | Балторо | Стандартный маршрут |
| 15 | Гашербрум II | 8035.00 | Пакистан | Балторо | Юго-западный склон |

Таблица Б.3 – Восхождения

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID восхождения | Дата начала | Дата окончания | Описание | Статус | ID горы | Целевая точка |
| 1 | 2023-07-15 | 2023-07-22 | Восхождение с юга, приют Бочки | выполнено | 1 | Западная вершина |
| 2 | 2023-08-10 | 2023-08-18 | Восхождение по маршруту Марангу | выполнено | 2 | Пик Ухуру |
| 3 | 2026-01-05 | 2026-01-12 | Зимнее восхождение с красивыми видами на гору | запланировано | 3 | Главная вершина |
| 4 | 2023-12-01 | 2023-12-20 | Экспедиция на Аконкагуа | не удалось | 4 | Главная вершина |
| 5 | 2026-04-10 | 2026-05-30 | Экспедиция на Эверест | запланировано | 5 | Главная вершина |
| 6 | 2024-07-10 | 2024-07-25 | Экспедиция на Макалу | выполнено | 6 | Главная вершина |
| 7 | 2024-08-05 | 2024-08-20 | Восхождение на Чогори | не удалось | 7 | Главная вершина |
| 8 | 2025-01-10 | 2025-01-25 | Зимнее восхождение на Лхоцзе | запланировано | 8 | Главная вершина |
| 9 | 2025-05-15 | 2025-06-10 | Экспедиция на Дхаулагири | выполнено | 9 | Главная вершина |
| 10 | 2026-03-01 | 2026-03-20 | Весеннее восхождение на Манаслу | запланировано | 10 | Главная вершина |
| 11 | 2024-09-05 | 2024-09-25 | Экспедиция на Нанга Парбат | не удалось | 11 | Главная вершина |
| 12 | 2025-04-10 | 2025-05-05 | Восхождение на Аннапурну | выполнено | 12 | Главная вершина |
| 13 | 2026-06-15 | 2026-07-10 | Летнее восхождение на Гашербрум I | запланировано | 13 | Главная вершина |
| 14 | 2024-10-01 | 2024-10-20 | Осенняя экспедиция на Броуд-Пик | выполнено | 14 | Главная вершина |
| 15 | 2025-02-15 | 2025-03-10 | Зимнее восхождение на Гашербрум II | выполнено | 15 | Главная вершина |

Таблица Б.4 – Заявки

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID заявки | ID участника | Дата подачи | Роль | Статус | ID восхождения |
| 1 | 1 | 2023-05-01 | участник | одобрено | 1 |
| 2 | 2 | 2023-05-05 | участник | одобрено | 1 |
| 3 | 1 | 2023-06-15 | участник | одобрено | 2 |
| 4 | 3 | 2023-11-20 | участник | на рассмотрении | 3 |
| 5 | 1 | 2024-02-01 | участник | отклонено | 5 |
| 6 | 2 | 2023-11-25 | руководитель | отклонено | 3 |
| 7 | 3 | 2023-12-01 | участник | выполнено | 1 |
| 8 | 3 | 2025-06-23 | участник | на рассмотрении | 5 |
| 9 | 3 | 2025-06-23 | участник | на рассмотрении | 3 |
| 14 | 5 | 2025-06-24 | участник | на рассмотрении | 5 |
| 15 | 2 | 2025-06-25 | участник | на рассмотрении | 5 |
| 16 | 7 | 2024-06-01 | участник | одобрено | 6 |
| 17 | 8 | 2024-06-05 | участник | одобрено | 7 |
| 18 | 9 | 2024-06-10 | участник | одобрено | 8 |
| 19 | 10 | 2024-06-15 | руководитель | одобрено | 9 |
| 20 | 11 | 2024-06-20 | участник | отклонено | 10 |
| 21 | 12 | 2024-07-01 | участник | одобрено | 11 |
| 22 | 13 | 2024-07-05 | участник | на рассмотрении | 12 |
| 23 | 14 | 2024-07-10 | участник | одобрено | 13 |
| 24 | 15 | 2024-07-15 | участник | отклонено | 14 |
| 25 | 16 | 2024-07-20 | участник | одобрено | 15 |
| 26 | 17 | 2024-08-01 | участник | на рассмотрении | 6 |
| 27 | 18 | 2024-08-05 | участник | одобрено | 7 |
| 28 | 7 | 2025-06-20 | участник | на рассмотрении | 8 |
| 29 | 8 | 2025-06-22 | участник | на рассмотрении | 9 |
| 30 | 9 | 2025-06-24 | руководитель | на рассмотрении | 10 |
| 1 | 1 | 2023-05-01 | участник | одобрено | 1 |
| 2 | 2 | 2023-05-05 | участник | одобрено | 1 |
| 3 | 1 | 2023-06-15 | участник | одобрено | 2 |
| 4 | 3 | 2023-11-20 | участник | на рассмотрении | 3 |

Код программы

Форма MounCMain.xaml.cs (Главное окно)

using System;

using System.Collections.Generic;

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 MountaineeringClub.View

{

/// <summary>

/// Логика взаимодействия для MounCMain.xaml

/// </summary>

public partial class MounCMain : Window

{

public MounCMain()

{

InitializeComponent();

LoadDatas();

}

private void LoadDatas()

{

if (AscentList != null)

AscentList.Items.Clear();

AscentList.Items.Add(new NewsControl());

}

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

{

new LoginWindow().Show();

this.Close();

}

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

{

new RegisterWindow().Show();

this.Close();

}

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

{

new MountainsWindow().Show(); this.Close();

}

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

{

new ExpeditionsWindow().Show(); this.Close();

}

}

}

Форма ExpeditionsWindow.xaml.cs (Окно восхождений)

using Microsoft.EntityFrameworkCore;

using MountaineeringClub.Model;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Windows;

using System.Windows.Controls;

namespace MountaineeringClub.View

{

public partial class ExpeditionsWindow : Window

{

private List<Ascent> \_allAscents;

public ExpeditionsWindow()

{

InitializeComponent();

LoadAscents();

}

private void LoadAscents()

{

try

{

var db = new DbMountaineeringClubContext();

db.Ascents

.Include(a => a.Mountain)

.Include(a => a.Applications)

.Include(a => a.Photographs)

.Load();

\_allAscents = db.Ascents.Local.ToList();

ExpeditionsDataGrid.ItemsSource = \_allAscents.Select(a => new

{

a.AscentId,

PhotoBytes = a.Photographs.FirstOrDefault()?.Image,

MountainName = a.Mountain?.Name,

TargetPoint = a.TargetPoint,

a.StartDate,

a.EndDate,

ParticipantsCount = a.Applications.Count,

AscentStatus = a.AscentStatus

}).ToList();

UpdateStatusText(\_allAscents.Count);

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка при загрузке восхождений: {ex.Message}", "Ошибка",

MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void UpdateStatusText(int count)

{

StatusText.Text = $"Загружено восхождений: {count}";

}

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

{

new MounCMain().Show();

this.Close();

}

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

{

var filteredAscents = \_allAscents.AsEnumerable();

if (DateFromPicker.SelectedDate.HasValue)

{

filteredAscents = filteredAscents.Where(a =>

a.StartDate >= DateOnly.FromDateTime(DateFromPicker.SelectedDate.Value));

}

if (DateToPicker.SelectedDate.HasValue)

{

filteredAscents = filteredAscents.Where(a =>

a.EndDate <= DateOnly.FromDateTime(DateToPicker.SelectedDate.Value));

}

if (StatusFilterComboBox.SelectedItem is ComboBoxItem selectedStatus &&

!string.IsNullOrEmpty(selectedStatus.Tag?.ToString()))

{

filteredAscents = filteredAscents.Where(a =>

a.AscentStatus == selectedStatus.Content.ToString());

}

ExpeditionsDataGrid.ItemsSource = filteredAscents.Select(a => new

{

a.AscentId,

PhotoBytes = a.Photographs.FirstOrDefault()?.Image,

MountainName = a.Mountain?.Name,

TargetPoint = a.TargetPoint,

a.StartDate,

a.EndDate,

ParticipantsCount = a.Applications.Count,

AscentStatus = a.AscentStatus

}).ToList();

UpdateStatusText(filteredAscents.Count());

}

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

{

DateFromPicker.SelectedDate = null;

DateToPicker.SelectedDate = null;

StatusFilterComboBox.SelectedIndex = -1;

LoadAscents();

}

}

}

Форма MountainsWindow.xaml.cs (Окно гор)

using Microsoft.EntityFrameworkCore;

using MountaineeringClub.Model;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Input;

namespace MountaineeringClub.View

{

public partial class MountainsWindow : Window

{

private List<Mountain> \_allMountains;

public MountainsWindow()

{

InitializeComponent();

LoadMountains();

DataContext = this;

}

private void LoadMountains()

{

try

{

var db = new DbMountaineeringClubContext();

db.Mountains

.Include(m => m.Ascents)

.Load();

\_allMountains = db.Mountains.Local.ToList();

MountainsDataGrid.ItemsSource = \_allMountains;

UpdateStatusText(\_allMountains.Count);

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка при загрузке гор: {ex.Message}", "Ошибка",

MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void UpdateStatusText(int count)

{

StatusText.Text = $"Загружено гор: {count}";

}

private void ApplyFilters()

{

var filteredMountains = \_allMountains.AsEnumerable();

if (!string.IsNullOrWhiteSpace(NameFilterTextBox.Text))

{

filteredMountains = filteredMountains.Where(m =>

m.Name.Contains(NameFilterTextBox.Text, StringComparison.OrdinalIgnoreCase));

}

if (!string.IsNullOrWhiteSpace(CountryFilterTextBox.Text))

{

filteredMountains = filteredMountains.Where(m =>

m.Country != null && m.Country.Contains(CountryFilterTextBox.Text, StringComparison.OrdinalIgnoreCase));

}

if (decimal.TryParse(MinHeightTextBox.Text, out decimal minHeight))

{

filteredMountains = filteredMountains.Where(m => m.Height >= minHeight);

}

if (decimal.TryParse(MaxHeightTextBox.Text, out decimal maxHeight))

{

filteredMountains = filteredMountains.Where(m => m.Height <= maxHeight);

}

MountainsDataGrid.ItemsSource = filteredMountains.ToList();

UpdateStatusText(filteredMountains.Count());

}

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

{

new MounCMain().Show();

this.Close();

}

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

{

NameFilterTextBox.Text = string.Empty;

CountryFilterTextBox.Text = string.Empty;

MinHeightTextBox.Text = string.Empty;

MaxHeightTextBox.Text = string.Empty;

MountainsDataGrid.ItemsSource = \_allMountains;

UpdateStatusText(\_allMountains.Count);

}

private void RussianText\_PreviewTextInput(object sender, TextCompositionEventArgs e)

{

foreach (var ch in e.Text)

{

if (!(ch >= 'А' && ch <= 'я') && ch != ' ' && ch != '-')

{

e.Handled = true;

return;

}

}

}

private void Number\_PreviewTextInput(object sender, TextCompositionEventArgs e)

{

foreach (var ch in e.Text)

{

if (!char.IsDigit(ch))

{

e.Handled = true;

return;

}

}

}

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

{

ApplyFilters();

}

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

{

ApplyFilters();

}

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

{

ApplyFilters();

}

}

}

Форма LoginWindow.xaml.cs (Окно авторизации)

using MountaineeringClub.Model;

using System;

using System.Collections.Generic;

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 MountaineeringClub.View

{

/// <summary>

/// Логика взаимодействия для LoginWindow.xaml

/// </summary>

public partial class LoginWindow : Window

{

public LoginWindow()

{

InitializeComponent();

}

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

{

new RegisterWindow().Show();

this.Close();

}

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

{

if (string.IsNullOrWhiteSpace(LoginTextBox.Text) || PasswordBox.Password == null)

{

MessageBox.Show("Пожалуйста, введите логин и пароль", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

return;

}

using (var db = new DbMountaineeringClubContext())

{

var userInDb = db.Users

.FirstOrDefault(u => u.EmailAddress == LoginTextBox.Text || u.ContactPhone == LoginTextBox.Text);

if (userInDb == null || userInDb.Password != PasswordBox.Password)

{

MessageBox.Show("Неверный логин или пароль", "Ошибка входа", MessageBoxButton.OK, MessageBoxImage.Error);

return;

}

if (userInDb.Role == "администратор")

{

new AdminWindow().Show();

this.Close();

return;

}

var participant = db.Participants.Where(p => p.UserId == userInDb.UserId).FirstOrDefault();

if (participant.MembershipStatus == "активный")

{

new UserProfileWindow(userInDb).Show();

this.Close();

}

else

{

MessageBox.Show("Участник временно не состоит в клубе", "Ошибка входа", MessageBoxButton.OK, MessageBoxImage.Error);

return;

}

}

}

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

{

new MounCMain().Show();

Close();

}

}

}

Форма Register.xaml.cs (Окно регистрации)

using System;

using System.Text.RegularExpressions;

using System.Windows;

using MountaineeringClub.Model;

using System.Linq;

using System.Windows.Controls;

using System.Windows.Media;

namespace MountaineeringClub.View

{

public partial class RegisterWindow : Window

{

private DbMountaineeringClubContext db;

public RegisterWindow()

{

InitializeComponent();

db = new DbMountaineeringClubContext();

BirthDatePicker.DisplayDateEnd = DateTime.Today.AddYears(-14);

}

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

{

new LoginWindow().Show();

this.Close();

}

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

{

ResetErrorMessages();

if (!ValidateFields())

return;

if (db.Users.Any(u => u.EmailAddress == EmailTextBox.Text))

{

ShowError(EmailErrorText, "Этот email уже зарегистрирован");

return;

}

if (db.Users.Any(u => u.ContactPhone == PhoneTextBox.Text))

{

ShowError(PhoneTextBox, "Этот номер телефона уже используется");

return;

}

DateTime? selectedDate = BirthDatePicker.SelectedDate;

if (!selectedDate.HasValue)

{

MessageBox.Show("Пожалуйста, выберите дату рождения", "Ошибка",

MessageBoxButton.OK, MessageBoxImage.Warning);

return;

}

var newUser = new User

{

Surname = LastNameTextBox.Text.Trim(),

Name = FirstNameTextBox.Text.Trim(),

Patronymic = MiddleNameTextBox.Text.Trim(),

BirthDate = DateOnly.FromDateTime(selectedDate.Value),

EmailAddress = EmailTextBox.Text.Trim(),

ContactPhone = PhoneTextBox.Text.Trim(),

RegistrationDate = DateOnly.FromDateTime(DateTime.Now),

Role = "зарегистрированный участник",

Password = PasswordBox.Password

};

db.Users.Add(newUser);

db.SaveChanges();

var newParticipant = new Participant

{

UserId = newUser.UserId,

MembershipStatus = "активный",

AscentHistory = ""

};

db.Participants.Add(newParticipant);

db.SaveChanges();

MessageBox.Show("Регистрация успешно завершена!", "Успех",

MessageBoxButton.OK, MessageBoxImage.Information);

new LoginWindow().Show();

this.Close();

}

private bool ValidateFields()

{

bool isValid = true;

if (string.IsNullOrWhiteSpace(LastNameTextBox.Text))

{

ShowError(LastNameTextBox, "Введите фамилию");

isValid = false;

}

if (string.IsNullOrWhiteSpace(FirstNameTextBox.Text))

{

ShowError(FirstNameTextBox, "Введите имя");

isValid = false;

}

if (string.IsNullOrWhiteSpace(MiddleNameTextBox.Text))

{

ShowError(MiddleNameTextBox, "Введите отчество");

isValid = false;

}

if (BirthDatePicker.SelectedDate == null)

{

MessageBox.Show("Укажите дату рождения", "Ошибка",

MessageBoxButton.OK, MessageBoxImage.Warning);

isValid = false;

}

else if (BirthDatePicker.SelectedDate > DateTime.Today.AddYears(-14))

{

MessageBox.Show("Минимальный возраст для регистрации - 14 лет", "Ошибка",

MessageBoxButton.OK, MessageBoxImage.Warning);

isValid = false;

}

if (string.IsNullOrWhiteSpace(EmailTextBox.Text))

{

ShowError(EmailTextBox, "Введите email");

isValid = false;

}

else if (!IsValidEmail(EmailTextBox.Text))

{

ShowError(EmailErrorText, "Некорректный формат email");

isValid = false;

}

if (string.IsNullOrEmpty(PasswordBox.Password))

{

ShowError(PasswordErrorText, "Введите пароль");

isValid = false;

}

else if (PasswordBox.Password.Length < 6)

{

ShowError(PasswordErrorText, "Пароль должен содержать минимум 6 символов");

isValid = false;

}

if (PasswordBox.Password != ConfirmPasswordBox.Password)

{

ShowError(PasswordErrorText, "Пароли не совпадают");

isValid = false;

}

if (string.IsNullOrWhiteSpace(PhoneTextBox.Text))

{

ShowError(PhoneTextBox, "Введите номер телефона");

isValid = false;

}

else if (!IsValidPhone(PhoneTextBox.Text))

{

ShowError(PhoneTextBox, "Некорректный формат телефона");

isValid = false;

}

return isValid;

}

private void ResetErrorMessages()

{

EmailErrorText.Visibility = Visibility.Collapsed;

PasswordErrorText.Visibility = Visibility.Collapsed;

LastNameTextBox.ClearValue(TextBox.BorderBrushProperty);

FirstNameTextBox.ClearValue(TextBox.BorderBrushProperty);

MiddleNameTextBox.ClearValue(TextBox.BorderBrushProperty);

EmailTextBox.ClearValue(TextBox.BorderBrushProperty);

PhoneTextBox.ClearValue(TextBox.BorderBrushProperty);

}

private void ShowError(TextBlock errorTextBlock, string message)

{

errorTextBlock.Text = message;

errorTextBlock.Visibility = Visibility.Visible;

}

private void ShowError(TextBox textBox, string message)

{

textBox.BorderBrush = Brushes.Red;

textBox.ToolTip = message;

}

private bool IsValidEmail(string email)

{

try

{

var addr = new System.Net.Mail.MailAddress(email);

return addr.Address == email;

}

catch

{

return false;

}

}

private bool IsValidPhone(string phone)

{

return Regex.IsMatch(phone, @"^[\d\s\(\)\-+]{10,}$");

}

}

}

Форма AdminWindow.xaml.cs (Окно администратора)

using Microsoft.EntityFrameworkCore;

using MountaineeringClub.Model;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.InteropServices;

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;

using iText.Kernel.Pdf;

using iText.Layout;

using Microsoft.Win32;

using iText.Layout.Element;

using iText.Layout.Properties;

using iText.Kernel.Colors;

using System.IO;

using iText.Kernel.Geom;

using System.Data;

using iText.IO.Font;

using iText.Kernel.Font;

namespace MountaineeringClub.View

{

/// <summary>

/// Логика взаимодействия для AdminWindow.xaml

/// </summary>

public partial class AdminWindow : Window

{

private readonly DbMountaineeringClubContext db;

public AdminWindow()

{

InitializeComponent();

db = new DbMountaineeringClubContext();

LoadData();

ApplicationsGrid.BeginningEdit += ApplicationsGrid\_BeginningEdit;

AscentsGrid.BeginningEdit += AscentsGrid\_BeginningEdit;

ReportTypeComboBox.SelectedIndex = 0;

}

private void LoadData()

{

db.Applications

.Include(a => a.Participant)

.Include(a => a.Participant.User)

.Include(a => a.Ascent)

.Include(a => a.Ascent.Mountain)

.Load();

ApplicationsGrid.ItemsSource = db.Applications.Local.ToObservableCollection();

db.Ascents

.Include(a => a.Mountain)

.Load();

AscentsGrid.ItemsSource = db.Ascents.Local.ToObservableCollection();

}

private void ApplicationsGrid\_BeginningEdit(object sender, DataGridBeginningEditEventArgs e)

{

if (e.Column.Header.ToString() != "Статус")

{

e.Cancel = true;

return;

}

var application = e.Row.Item as Model.Application;

if (application != null && application.ApplicationStatus == "выполнено")

{

e.Cancel = true;

MessageBox.Show("Нельзя изменить статус 'выполнено'", "Ошибка",

MessageBoxButton.OK, MessageBoxImage.Warning);

}

}

private void AscentsGrid\_BeginningEdit(object sender, DataGridBeginningEditEventArgs e)

{

if (e.Column.Header.ToString() != "Статус")

{

e.Cancel = true;

return;

}

var ascent = e.Row.Item as Ascent;

if (ascent != null && ascent.AscentStatus == "выполнено")

{

e.Cancel = true;

MessageBox.Show("Нельзя изменить статус 'выполнено'", "Ошибка",

MessageBoxButton.OK, MessageBoxImage.Warning);

}

}

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

{

try

{

var changedEntries = db.ChangeTracker.Entries()

.Where(entry => entry.State == EntityState.Modified)

.ToList();

bool hasCompletedChanges = changedEntries.Any(entry =>

{

if (entry.Entity is Model.Application app)

return entry.OriginalValues["ApplicationStatus"].ToString() == "выполнено";

else if (entry.Entity is Ascent ascent)

return entry.OriginalValues["AscentStatus"].ToString() == "выполнено";

return false;

});

if (hasCompletedChanges)

{

MessageBox.Show("Нельзя изменять записи со статусом 'выполнено'", "Ошибка",

MessageBoxButton.OK, MessageBoxImage.Error);

return;

}

if (changedEntries.Any())

{

db.SaveChanges();

MessageBox.Show("Изменения сохранены", "Успех",

MessageBoxButton.OK, MessageBoxImage.Information);

}

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка при сохранении: {ex.Message}", "Ошибка",

MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void GenerateMountainStatistics()

{

var mountainStats = db.Ascents

.Include(a => a.Mountain)

.GroupBy(a => a.Mountain.Name)

.Select(g => new

{

Гора = g.Key,

КоличествоВосхождений = g.Count(),

ПервоеВосхождение = g.Min(a => a.StartDate),

ПоследнееВосхождение = g.Max(a => a.StartDate)

})

.OrderByDescending(x => x.КоличествоВосхождений)

.ThenBy(x => x.Гора)

.ToList();

DataTable table = new DataTable();

table.Columns.Add("Гора", typeof(string));

table.Columns.Add("Количество восхождений", typeof(int));

table.Columns.Add("Первое восхождение", typeof(string));

table.Columns.Add("Последнее восхождение", typeof(string));

foreach (var stat in mountainStats)

{

table.Rows.Add(

stat.Гора,

stat.КоличествоВосхождений,

stat.ПервоеВосхождение?.ToString("dd.MM.yyyy") ?? "н/д",

stat.ПоследнееВосхождение?.ToString("dd.MM.yyyy") ?? "н/д"

);

}

ReportsGrid.ItemsSource = table.DefaultView;

}

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

{

try

{

var selectedReport = (ReportTypeComboBox.SelectedItem as ComboBoxItem)?.Content.ToString();

switch (selectedReport)

{

case "Статистика по горам (по количеству восхождений)":

GenerateMountainStatistics();

break;

case "Рейтинг участников (по количеству восхождений)":

GenerateParticipantRatings();

break;

case "Отчет по завершенным восхождениям (по дате)":

GenerateCompletedAscentsReport();

break;

default:

MessageBox.Show("Выберите тип отчета", "Ошибка",

MessageBoxButton.OK, MessageBoxImage.Warning);

break;

}

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка при формировании отчета: {ex.Message}", "Ошибка",

MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void GenerateParticipantRatings()

{

var participantRatings = db.Applications

.Where(a => a.ApplicationStatus == "выполнено")

.Select(a => new

{

UserId = a.Participant.User.UserId,

Surname = a.Participant.User.Surname,

Name = a.Participant.User.Name,

Patronymic = a.Participant.User.Patronymic,

StartDate = a.Ascent.StartDate

})

.AsEnumerable()

.GroupBy(a => new

{

a.UserId,

FullName = $"{a.Surname} {a.Name} {a.Patronymic}"

})

.Select(g => new

{

Участник = g.Key.FullName,

КоличествоВосхождений = g.Count(),

ПервоеВосхождение = g.Min(a => a.StartDate),

ПоследнееВосхождение = g.Max(a => a.StartDate)

})

.OrderByDescending(x => x.КоличествоВосхождений)

.ToList();

// Создаем DataTable

DataTable table = new DataTable();

table.Columns.Add("Участник", typeof(string));

table.Columns.Add("Количество восхождений", typeof(int));

table.Columns.Add("Первое восхождение", typeof(string));

table.Columns.Add("Последнее восхождение", typeof(string));

foreach (var rating in participantRatings)

{

table.Rows.Add(

rating.Участник,

rating.КоличествоВосхождений,

rating.ПервоеВосхождение?.ToString("dd.MM.yyyy") ?? "н/д",

rating.ПоследнееВосхождение?.ToString("dd.MM.yyyy") ?? "н/д"

);

}

ReportsGrid.ItemsSource = table.DefaultView;

}

private void GenerateCompletedAscentsReport()

{

var completedAscents = db.Ascents

.Include(a => a.Mountain)

.Where(a => a.AscentStatus == "выполнено")

.OrderByDescending(a => a.StartDate)

.Select(a => new

{

Гора = a.Mountain.Name,

Высота = a.Mountain.Height,

Дата = a.StartDate,

Цель = a.TargetPoint,

Продолжительность = a.StartDate.HasValue && a.EndDate.HasValue

? (a.EndDate.Value.DayNumber - a.StartDate.Value.DayNumber + 1)

: (int?)null

})

.ToList();

// Создаем DataTable

DataTable table = new DataTable();

table.Columns.Add("Гора", typeof(string));

table.Columns.Add("Высота", typeof(decimal));

table.Columns.Add("Дата", typeof(string));

table.Columns.Add("Цель", typeof(string));

table.Columns.Add("Продолжительность", typeof(string));

foreach (var ascent in completedAscents)

{

table.Rows.Add(

ascent.Гора,

ascent.Высота,

ascent.Дата?.ToString("dd.MM.yyyy") ?? "н/д",

ascent.Цель,

ascent.Продолжительность?.ToString() ?? "н/д"

);

}

ReportsGrid.ItemsSource = table.DefaultView;

}

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

{

try

{

if (ReportsGrid.Items.Count == 0)

{

MessageBox.Show("Нет данных для экспорта", "Ошибка",

MessageBoxButton.OK, MessageBoxImage.Warning);

return;

}

string fontPath = System.IO.Path.Combine(Environment.GetFolderPath(Environment.SpecialFolder.Fonts), "arial.ttf");

if (!File.Exists(fontPath))

{

fontPath = System.IO.Path.Combine(AppDomain.CurrentDomain.BaseDirectory, "Fonts", "times.ttf");

}

var saveFileDialog = new SaveFileDialog

{

Filter = "PDF files (\*.pdf)|\*.pdf",

FileName = $"Отчет\_{DateTime.Now:yyyyMMdd}.pdf"

};

if (saveFileDialog.ShowDialog() == true)

{

DataView dataView = (DataView)ReportsGrid.ItemsSource;

DataTable table = dataView.Table;

using (var writer = new PdfWriter(saveFileDialog.FileName))

using (var pdf = new PdfDocument(writer))

{

PdfFont font;

try

{

font = PdfFontFactory.CreateFont(fontPath, PdfEncodings.IDENTITY\_H, PdfFontFactory.EmbeddingStrategy.PREFER\_EMBEDDED);

}

catch

{

font = PdfFontFactory.CreateFont("Helvetica", PdfEncodings.IDENTITY\_H);

}

var document = new Document(pdf, PageSize.A4.Rotate());

document.SetMargins(20, 20, 20, 20);

iText.Layout.Style russianStyle = new iText.Layout.Style()

.SetFont(font)

.SetFontSize(12);

document.Add(new iText.Layout.Element.Paragraph("Отчет")

.SetTextAlignment(iText.Layout.Properties.TextAlignment.CENTER)

.SetFontSize(16)

.AddStyle(russianStyle));

document.Add(new iText.Layout.Element.Paragraph($"Дата формирования: {DateTime.Now:dd.MM.yyyy HH:mm}")

.SetTextAlignment(iText.Layout.Properties.TextAlignment.RIGHT)

.SetFontSize(10)

.AddStyle(russianStyle));

var pdfTable = new iText.Layout.Element.Table(table.Columns.Count)

.UseAllAvailableWidth();

iText.Layout.Style headerStyle = new iText.Layout.Style()

.SetBackgroundColor(new DeviceRgb(51, 102, 153))

.SetFontColor(ColorConstants.WHITE)

.SetTextAlignment(iText.Layout.Properties.TextAlignment.CENTER)

.SetFont(font);

foreach (DataColumn column in table.Columns)

{

pdfTable.AddHeaderCell(new Cell()

.Add(new iText.Layout.Element.Paragraph(column.ColumnName)

.AddStyle(headerStyle)));

}

iText.Layout.Style cellStyle = new iText.Layout.Style()

.SetFont(font)

.SetTextAlignment(iText.Layout.Properties.TextAlignment.CENTER);

foreach (DataRow row in table.Rows)

{

foreach (DataColumn column in table.Columns)

{

pdfTable.AddCell(new Cell()

.Add(new iText.Layout.Element.Paragraph(row[column].ToString())

.AddStyle(cellStyle)));

}

}

document.Add(pdfTable);

document.Add(new iText.Layout.Element.Paragraph("Сгенерировано системой управления альпинистским клубом")

.SetTextAlignment(iText.Layout.Properties.TextAlignment.CENTER)

.SetFontSize(10)

.AddStyle(russianStyle));

}

MessageBox.Show("Отчет успешно экспортирован в PDF", "Успех",

MessageBoxButton.OK, MessageBoxImage.Information);

}

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка при экспорте в PDF: {ex.Message}", "Ошибка",

MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void TabControl\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

if (sender is TabControl tabControl)

{

if (tabControl.SelectedIndex == 2)

{

ButtonsPanel.Visibility = Visibility.Collapsed;

}

else

{

ButtonsPanel.Visibility = Visibility.Visible;

}

}

}

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

{

new MounCMain().Show();

this.Close();

}

}

}

Форма UserProfileWindow.xaml.cs (Окно пользователя)

using Microsoft.EntityFrameworkCore;

using MountaineeringClub.Model;

using MountaineeringClub.UserControls;

using System;

using System.Collections.Generic;

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 MountaineeringClub.View

{

/// <summary>

/// Логика взаимодействия для UserProfileWindow.xaml

/// </summary>

public partial class UserProfileWindow : Window

{

private Model.User current\_user;

private Participant current\_participant;

public UserProfileWindow(User user)

{

InitializeComponent();

current\_user = user;

DataContext = current\_user;

LoadDatas();

}

private void LoadDatas()

{

using (var db = new DbMountaineeringClubContext())

{

current\_participant = db.Participants

.Where(p => p.UserId == current\_user.UserId).FirstOrDefault();

if (ApplicationsListView != null)

ApplicationsListView.Items.Clear();

ApplicationsListView.Items.Add(new ApplicationsHistoryControl(current\_participant.ParticipantId));

if (AscentsHistoryList != null)

AscentsHistoryList.Items.Clear();

AscentsHistoryList.Items.Add(new AscentsHistoryControl(current\_participant.ParticipantId));

int activeApplications = GetActiveApplicationsCount(current\_participant.ParticipantId);

int activeAscents = GetActiveAscentsCount(current\_participant.ParticipantId);

noApplications\_msg.Visibility = activeApplications == 0

? Visibility.Visible

: Visibility.Collapsed;

noAscents\_msg.Visibility = activeAscents == 0

? Visibility.Visible

: Visibility.Collapsed;

}

}

private int GetActiveApplicationsCount(int participantId)

{

using (var db = new DbMountaineeringClubContext())

{

return db.Applications

.Where(a => a.ParticipantId == participantId &&

(a.Ascent == null || a.Ascent.AscentStatus != "выполнено"))

.Count();

}

}

private int GetActiveAscentsCount(int participantId)

{

using (var db = new DbMountaineeringClubContext())

{

return db.Applications

.Include(a => a.Participant)

.Where(a => a.ApplicationStatus == "выполнено" &&

a.ParticipantId == participantId).Count();

}

}

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

{

new MounCMain().Show();

this.Close();

}

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

{

var newAppWindow = new NewApplicationWindow(current\_user);

newAppWindow.Owner = this;

if (newAppWindow.ShowDialog() == true)

{

LoadDatas();

}

}

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

{

SwitchToViewMode();

}

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

{

NewApplicationButton.IsEnabled = false;

UserNameText.Visibility = Visibility.Collapsed;

UserSPText.Visibility = Visibility.Collapsed;

UserEmailText.Visibility = Visibility.Collapsed;

UserPhoneText.Visibility = Visibility.Collapsed;

UserRegDateText.Visibility = Visibility.Collapsed;

LeaveTheClubButton.Visibility = Visibility.Collapsed;

EditFieldsPanel.Visibility = Visibility.Visible;

EditProfileButton.Visibility = Visibility.Collapsed;

EditButtonsPanel.Visibility = Visibility.Visible;

SurnameTextBox.Text = current\_user.Surname;

NameTextBox.Text = current\_user.Name;

PatronymicTextBox.Text = current\_user.Patronymic;

EmailTextBox.Text = current\_user.EmailAddress;

PhoneTextBox.Text = current\_user.ContactPhone;

}

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

{

if (string.IsNullOrWhiteSpace(SurnameTextBox.Text))

{

MessageBox.Show("Пожалуйста, укажите фамилию", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning);

SurnameTextBox.Focus();

return;

}

if (string.IsNullOrWhiteSpace(NameTextBox.Text))

{

MessageBox.Show("Пожалуйста, укажите имя", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning);

NameTextBox.Focus();

return;

}

if (string.IsNullOrWhiteSpace(PatronymicTextBox.Text))

{

MessageBox.Show("Пожалуйста, укажите отчество", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning);

SurnameTextBox.Focus();

return;

}

if (string.IsNullOrWhiteSpace(EmailTextBox.Text))

{

MessageBox.Show("Пожалуйста, укажите email", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning);

EmailTextBox.Focus();

return;

}

else if (!EmailTextBox.Text.Contains("@") || !EmailTextBox.Text.Contains("."))

{

MessageBox.Show("Пожалуйста, укажите корректный email", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning);

EmailTextBox.Focus();

return;

}

if (string.IsNullOrWhiteSpace(PhoneTextBox.Text))

{

MessageBox.Show("Пожалуйста, укажите телефон", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning);

PhoneTextBox.Focus();

return;

}

else if (PhoneTextBox.Text.Length < 10)

{

MessageBox.Show("Номер телефона должен содержать не менее 10 цифр", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning);

PhoneTextBox.Focus();

return;

}

try

{

using (var db = new DbMountaineeringClubContext())

{

var userInDb = db.Users.Find(current\_user.UserId);

userInDb.Surname = SurnameTextBox.Text.Trim();

userInDb.Name = NameTextBox.Text.Trim();

userInDb.Patronymic = PatronymicTextBox.Text.Trim();

userInDb.EmailAddress = EmailTextBox.Text.Trim();

userInDb.ContactPhone = PhoneTextBox.Text.Trim();

db.SaveChanges();

current\_user = userInDb;

DataContext = null;

DataContext = current\_user;

}

MessageBox.Show("Данные успешно сохранены!", "Успех", MessageBoxButton.OK, MessageBoxImage.Information);

SwitchToViewMode();

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка при сохранении: {ex.Message}", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void SwitchToViewMode()

{

EditFieldsPanel.Visibility = Visibility.Collapsed;

EditButtonsPanel.Visibility = Visibility.Collapsed;

NewApplicationButton.IsEnabled = true;

UserNameText.Visibility = Visibility.Visible;

UserSPText.Visibility = Visibility.Visible;

UserEmailText.Visibility = Visibility.Visible;

UserPhoneText.Visibility = Visibility.Visible;

UserRegDateText.Visibility = Visibility.Visible;

EditProfileButton.Visibility = Visibility.Visible;

LeaveTheClubButton.Visibility = Visibility.Visible;

}

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

{

var result = MessageBox.Show("Вы точно хотите покинуть клуб?",

"Подтверждение выхода",

MessageBoxButton.YesNo,

MessageBoxImage.Question);

if (result == MessageBoxResult.Yes)

{

try

{

using (var db = new DbMountaineeringClubContext())

{

var participant = db.Participants.FirstOrDefault(p => p.UserId == current\_user.UserId);

if (participant != null)

{

participant.MembershipStatus = "неактивный";

db.SaveChanges();

new MounCMain().Show();

this.Close();

}

else

{

MessageBox.Show("Ошибка: участник не найден",

"Ошибка",

MessageBoxButton.OK,

MessageBoxImage.Error);

}

}

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка при выходе из клуба: {ex.Message}",

"Ошибка",

MessageBoxButton.OK,

MessageBoxImage.Error);

}

}

}

}

}

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Потопахин В. Искусство алгоритмизации. [Текст] / В. Потопахин. – Москва: ДМК Пресс, 2018. – 320 с.
2. Дейтел П. C# для профессионалов: тонкости программирования. [Текст] / П. Дейтел, Х. Дейтел. – Москва: Лори, 2019. – 880 с.
3. Скиена С. Алгоритмы. Руководство по разработке [Текст] / С. Скиена 2-е изд. – СПб. БХВ-Петербург, 2016. – 720 с.
4. Троелсен Э. Язык программирования C# 9.0 и платформа .NET 5. [Текст] / Э. Троелсен. – Москва: Вильямс, 2021. – 1328 с.
5. Программирование на C, C# и Java [Электронный ресурс] / MySQL и C# работаем с базой данных. – Режим доступа https://vscode.ru/proglessons/mysql-c-sharp.html, свободный
6. Metanit [Электронный ресурс] / Руководство по языку C#. – Режим доступа https://metanit.com/sharp/tutorial/, свободный
7. Metanit [Электронный ресурс] / Руководство по WPF. – Режим доступа https://metanit.com/sharp/wpf/, свободный
8. Microsoft [Электронный ресурс] / Руководство по языку C#. – Режим доступа https://docs.microsoft.com/ru-ru/dotnet/csharp/, свободный
9. Professor Web [Электронный ресурс] / Основы WPF. – Режим доступа https://professorweb.ru/my/WPF/base\_WPF/level1/info\_WPF.php, свободный
10. Ruseller [Электронный ресурс] / Введение в хранимые процедуры MySQL 5. – Режим доступа https://ruseller.com/lessons.php?id=1189, свободный
11. Sql-tutorial [Электронный ресурс] / SQL Задачи и решения. – Режим доступа http://www.sql-tutorial.ru/ru/content.html, свободный
12. Официальная документация Microsoft по WPF и .NET [Электронный ресурс]. – Режим доступа: <https://docs.microsoft.com/ru-ru/dotnet/desktop/wpf/>, свободный
13. Webmasterwiki [Электронный ресурс] / Базы данных: Основные понятия. – Режим доступа https://www.webmasterwiki.ru/MySQL, свободный
14. WorldSkills Russia [Электронный ресурс] / Программные решения для бизнеса. – Режим доступа https://nationalteam.worldskills.ru/skills/programmnye-resheniya-dlyabiznesa/, свободный