**Оглавление**

[Аннотация 4](#_Toc26196310)

[Введение 5](#_Toc26196311)

[1 ОБЩИЕ СВЕДЕНИЯ О ПРОГРАММНОМ СРЕДСТВЕ 6](#_Toc26196312)

[1.1 Основное функциональное назначение программного средства 6](#_Toc26196313)

[1.2 Полное наименование программного средства 6](#_Toc26196314)

[1.3 Условное обозначение программного средства 6](#_Toc26196315)

[1.4 Разработчики программного средства 6](#_Toc26196316)

[2 ТЕХНИЧЕСКОЕ ЗАДАНИЕ 7](#_Toc26196317)

[2.1 Основание для разработки 7](#_Toc26196318)

[2.2 Назначение разработки 7](#_Toc26196319)

[2.3 Требование к программному средству 7](#_Toc26196320)

[2.4 Требования к программной документации 8](#_Toc26196321)

[2.5 Требования к эргономике и технической эстетике 8](#_Toc26196322)

[2.6 Стадии и этапы разработки 9](#_Toc26196323)

[2.7 Порядок контроля и приемки 10](#_Toc26196324)

[3 ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К ПРОГРАММНОМУ ПРОДУКТУ 11](#_Toc26196325)

[3.1 Декомпозиция поставленной задачи 11](#_Toc26196326)

[3.2 Общая архитектура программного средства 12](#_Toc26196327)

[3.3 Разработка алгоритма решения задачи 15](#_Toc26196328)

[3.4 Реализация функционального назначения программного средства 15](#_Toc26196329)

[3.5 Структурная организация данных 16](#_Toc26196330)

[3.6 Разработка интерфейса ПС 22](#_Toc26196331)

[3.7 Описание структуры выходной информации 30](#_Toc26196332)

[4 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ 32](#_Toc26196333)

[4.1 Назначение программного средства 32](#_Toc26196334)

[4.2 Условия выполнения программного средства 32](#_Toc26196335)

[4.3 Эксплуатация программного средства 32](#_Toc26196336)

[4.4 Сообщения пользователю 40](#_Toc26196337)

[5 РАЗРАБОТКА ТЕСТОВОГО НАБОРА 42](#_Toc26196338)

[5.1 Обоснование необходимого количества тестов 42](#_Toc26196339)

[5.2 Описание тестовых пакетов 43](#_Toc26196340)

[6. ЗАКЛЮЧЕНИЕ 55](#_Toc26196341)

[6.1 Оценка качества программного средства с помощью метрик 55](#_Toc26196342)

[7. СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 58](#_Toc26196343)

[Приложение А листинг программного модуля 59](#_Toc26196344)

[Приложение B диск с исполняемым кодом программного модуля 98](#_Toc26196345)

**Аннотация**

На курсовой проект студента Белова А.С. на тему: Разработка интегрированного программного модуля «Охотхозяйство ООО Охотничьи традиции». Курсовой проект представлен пояснительной запиской на 98 листов машинописного текста. Состоит из введения, 5 глав, заключения и списка литературных источников.

В главе 1 содержится общие сведения о программном средстве Охотхозяйство ООО Охотничьи традиции. В главе 2 приведено техническое задание на проектирование БД согласно ГОСТ 19.201. Глава 3 содержит пояснительную записку к программному продукту согласно ГОСТ 19.404-79. В главе 4 содержится руководство пользователя согласно ГОСТ 19.504-79. Глава 5 содержит обоснование количества необходимых тестов и тестовые пакеты для оценки работоспособности программного продукта. В главе 6 приведены результаты ручного и автоматического тестирования программного продукта и выполнена оценка качества с помощью метрик.

В работе содержится общее описание программного средства, предназначенного для объекта проектирования – интегрированного программного модуля «Охотхозяйство ООО Охотничьи традиции», изложены требования к программному средству, описаны его архитектура, реализация функций, интерфейс, приведено руководство пользователя, обоснованы тестовые пакеты и выполнена оценка качества программного продукта с помощью метрик.

**Введение**

В курсовом проекте была поставлена задача разработки интегрированного программного модуля «Охотхозяйство ООО Охотничьи традиции», который позволяет автоматизировать учет мероприятий.

Для достижения поставленной цели был составлен основной алгоритм работы программы, а также разработаны алгоритмы вспомогательных процедур.

Для реализации алгоритмов был выбран язык программирования C#. Выбор обусловлен тем, что возможностей этого языка достаточно для достижения поставленной цели моего курсового проекта.

При разработке интерфейса программы, я учитывал наиболее нужные и важные функциональные возможности, которые должна выполнять данная программа, а также тип пользователей, которые с ней будут работать.

**1. ОБЩИЕ СВЕДЕНИЯ О ПРОГРАММНОМ СРЕДСТВЕ**

**1.1 Основное функциональное назначение программного средства**

Программа предназначена для автоматизации учета мероприятий. Программа не требует владения современными информационными технологиями для эффективного использования данного программного средства, т.е. подходит для любого пользователя.

**1.2 Полное наименование программного средства**

Разработка интегрированного программного модуля «Охотхозяйство ООО Охотничьи традиции».

**1.3 Условное обозначение программного средства**

«Hunt.01»

**1.4 Разработчики программного средства**

Программное средство разработал студент Зеленодольского Механического колледжа группы 205 Белов А.С.

**2. ТЕХНИЧЕСКОЕ ЗАДАНИЕ**

**2.1 Основание для разработки**

Работа ведётся на основании задания на курсовой проект по специальности 09.02.03 — «Программирование в компьютерных системах».

**2.2 Назначение разработки**

Программа «Hunt.01» работает под управление операционной системы Windows 7, Windows 8, Windows 10. Программа предназначена для автоматизации деятельности администратора по учету мероприятий охотхозяйства. Программа позволяет вводить, изменять и удалять данные мероприятиях.

**2.3 Требование к программному средству**

«Hunt.01» устанавливается на персональный компьютер, имеющий представленные минимальные системные требования:

Оперативная память: 4 Гб;

Процессор: Intel Core i / Phenom II X4 / AMD Athlon64 X2 4600 + 2.4 ГГц;

Видео карта: nVidia GeForce 8600 / ATI Radeon HD 2400;

Звуковая карта: Совместимая с DirectX;

Программный продукт «Hunt.01» является кроссплатформенным продуктом, что позволяет не ограничиваться в работе с определенной версией ОС.

«Hunt.01» написана на объектно-ориентированном языке программировании C# (си-шарп) в интегрированной среде разработки Visual Studio 2019. В качестве СУБД используется MS SQL. Программа не может подлежать сторонней оптимизации и редактированию программного кода.

**2.4 Требования к программной документации**

Состав программной документации должен включать в себя:

1. техническое задание;

2. сроки выполнения работ;

3. программу и методику испытаний;

4. эксплуатационные инструкции пользователю;

**2.5 Требования к эргономике и технической эстетике**

Система обеспечивает удобный для пользователей системный интерфейс, отвечающий следующим требованиям:

- единый стиль оформления для пользовательских интерфейсов;

- удобная, интуитивно понятная навигация в интерфейсе пользователя;

- взаимодействие пользователя с системой осуществляется на двух языках по выбору (русский и английский), для наибольшего удобства

Пользовательские интерфейсы системы спроектированы и разработаны с применением единых принципов графического представления информации и организации доступа к функциональным возможностям и сервисам. Разработан графический дизайн пользовательских интерфейсов, цветовые, шрифтовые и композиционные решения для отображения текстов, изображений, таблиц, гиперссылок, управляющих и навигационных элементов (меню, кнопок, форм и т.п.).

Экранные формы спроектированы с учетом требований унификации:

- все экранные формы пользовательского интерфейса выполнены в едином графическом дизайне, с одинаковым расположением основных элементов управления и навигации;

- для обозначения сходных операций использованы сходные графические значки, кнопки и другие управляющие (навигационные) элементы. Термины, используемые для обозначения типовых операций (добавление информационной сущности, редактирование поля данных), а также последовательности действий пользователя при их выполнении унифицированы;

- внешнее поведение сходных элементов интерфейса (реакция на наведение указателя «мыши», переключение фокуса, нажатие кнопки) реализованы одинаково для однотипных элементов.

**2.6 Стадии и этапы разработки**

Разработка должна быть проведена в три стадии:

1. разработка технического задания;

2. рабочее проектирование;

3. внедрение.

На стадии разработки технического задания должен быть выполнен этап разработки, согласования и утверждения настоящего технического задания.

На стадии рабочего проектирования должны быть выполнены перечисленные ниже этапы работ:

1. разработка программы;

2. разработка программной документации;

3. испытания программы.

На этапе разработки технического задания должны быть выполнены перечисленные ниже работы:

1. постановка задачи;

2. определение и уточнение требований к техническим средствам;

3. определение требований к программе;

4. определение стадий, этапов и сроков разработки программы и документации на нее;

5. выбор языков программирования;

6. согласование и утверждение технического задания.

На этапе разработки программы должна быть выполнена работа по программированию (кодированию) и отладке программы.

На этапе разработки программной документации должна быть выполнена разработка программных документов в соответствии с пунктом Предварительный состав программной документации настоящего технического задания.

На этапе испытаний программы должны быть выполнены перечисленные ниже виды работ:

1. разработка, согласование и утверждение порядка и методики испытаний;

2. проведение испытаний;

3. корректировка программы и программной документации по результатам испытаний.

**2.7 Порядок контроля и приемки**

Сдача-приёмка работ производится поэтапно, в соответствии с рабочей программой и календарным планом. Сдача-приемка осуществляется комиссией, в состав которой входят представители ГАПОУ и работодатель. По результатам приемки подписывается акт приемочной комиссии.

В указанном подразделе, согласно таблице «Сроки выполнения работ» технического задания, будет описаны перечень участвующих организаций, место и сроки проведения работ, согласно п. 2.8 ГОСТ 34.602-89.

Порядок согласования и утверждения приемочной документации должен регламентироваться организационно-распорядительной документацией организации, принимающей участие в создании программного продукта. Согласно разделу «Приемка результатов разработки» ГОСТ 15.001-88 для согласования и утверждения приемочной документации создается приемочная комиссия (приказом).

Статус приемочной комиссии определяется Заказчиком до проведения испытаний.

**3. ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К ПРОГРАММНОМУ ПРОДУКТУ**

**3.1 Декомпозиция поставленной задачи**

Для достижения цели работы была поставлена следующая задача:

Создать модуль “CodeDatabase” отвечающий за сохранение БД. Результат работы модуля - сохранение БД.

Поставленные задачи наглядно отображаются на диаграмме деятельности.

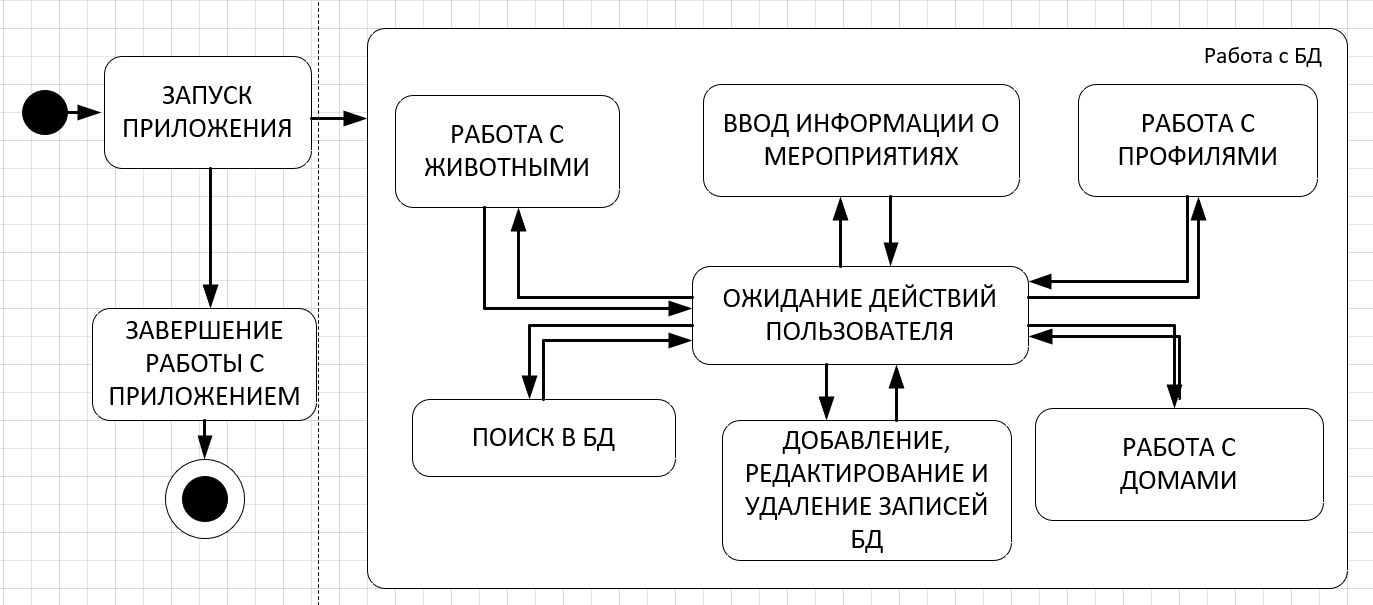


Рис.1 Декомпозиция поставленной задачи

**3.2 Общая архитектура программного средства**

Объектом моделирования является Охотхозяйство ООО Охотничьи традиции.

Рассматриваемые процессы: Добавление и удаление информации о мероприятиях, животных, домах, пользователях.

Объекты моделирования представлены на диаграмме классов, рис.2.

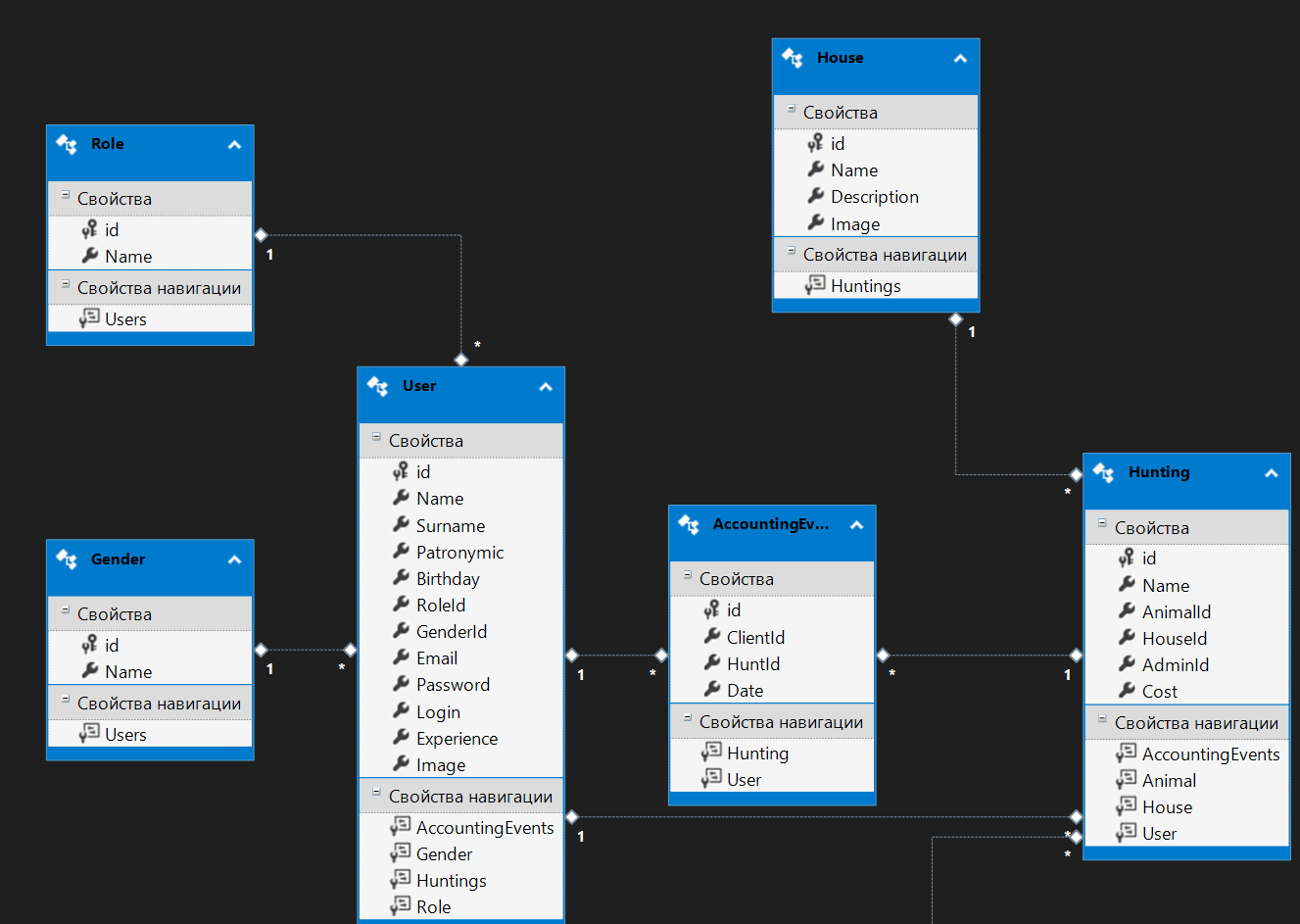


Рис.2 Диаграмма классов

По полученной декомпозиции задач была спроектирована следующая архитектура программного средства.

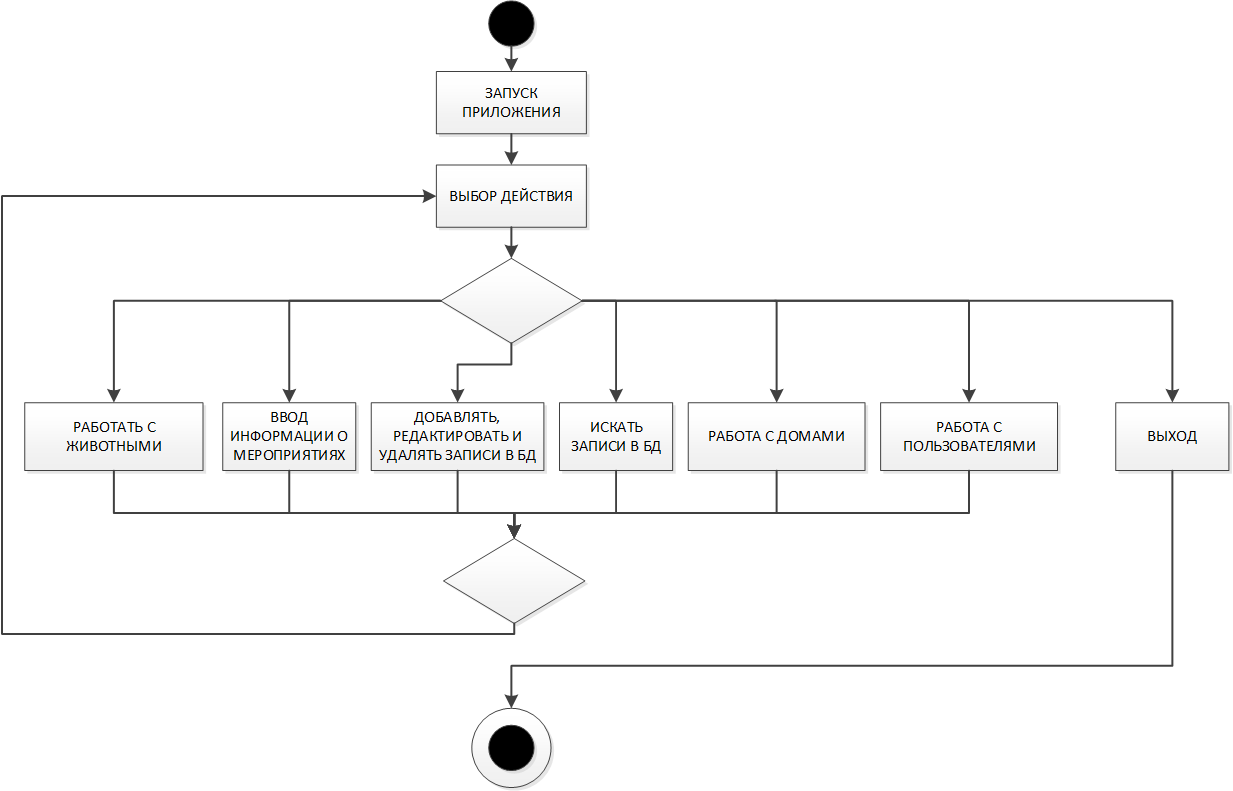


Рис.3 Схема событийно-ориентированной архитектуры

В «Hunt.01» используется событийно-ориентированная архитектура.(Рис.3) Роль администратора (источник событий) в программе выполняют: пользователи программы, за роль стоков (потребители событий) отвечают таблицы, входящие в базу данных. Например, когда пользователь выбирает определенное действие: добавить, удалить, редактировать, вывести в Excel и т.д. система осуществит выбранные действия, и база отреагирует соответствующим образом: запись добавлена, удалена, отредактирована, либо отчет выведен в Excel. Системная архитектура пользователя рассматривает это изменение состояния как событие, создаваемое, публикуемое, определяемое и потребляемое различными приложениями в составе архитектуры.

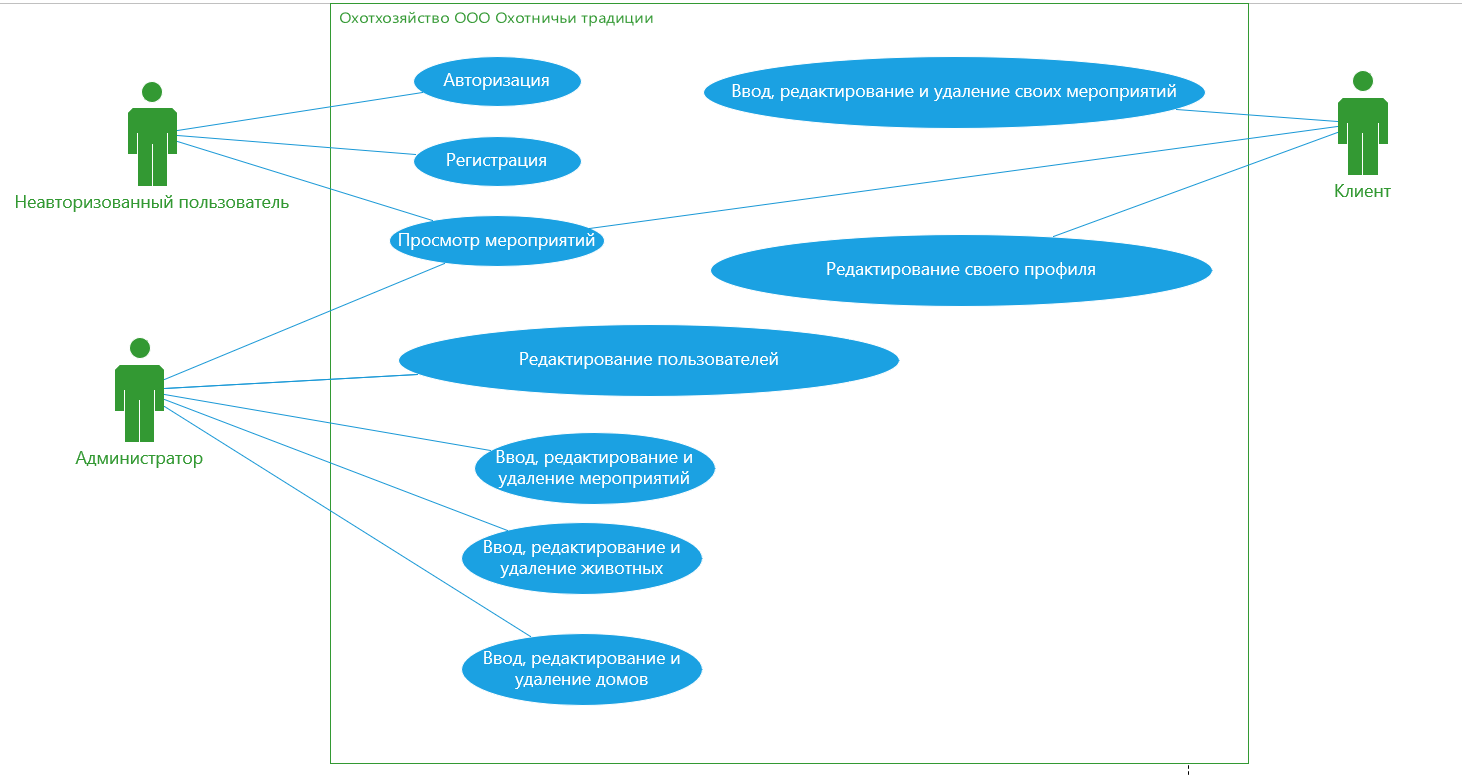


Рис.4 Диаграмма вариантов использования

**3.3 Разработка алгоритма решения задачи**

**Общий алгоритм**

Рис.5 Алгоритм программы

**3.4 Реализация функционального назначения программного средства**

Программа «Hunt.01» имеет следующий набор входных данных, такие как: мероприятия, животные, пользователи, дома и информация о них.

Данные вводятся администратором в соответствующие поля ввода, снабженные всплывающими подсказками.

Выходными данными являются: отображение списка мероприятий, животных, домов в виде плиточного интерфейса.

Данные вводятся пользователем в соответствующие поля вывода, снабженные всплывающими подсказками.

Выходные данные редактировать вручную возможно.

**3.5 Структурная организация данных**

Для создания БД необходимо определиться с данными, которые необходимы для полноценного функционирования системы. Все эти данные указаны в реляционной модели «БД Hunt» представленная на рисунке. Любая реляционная база данных и называется реляционной, что характеризуется отношениями (relation) между таблицами. На рисунке изображены таблицы моей базы данных. При этом четыре таблицы являются родительскими (главными), а другие – дочерними (подчиненными). Главными таблицами являются «Hunting», «User», «AccountingEvents», «Animal». Реляционная модель автоматизированной системы соответствует всем 12 правилам Кодда. (рис.6)



Рис.6 Реляционная модель

Первичный ключ в базе уникален, используется для организации отношений между таблицами, который не может иметь пустых и повторяющихся значений. Первичными ключами в базе являются поля: id (таблица Role), id (Таблица Difficulty), id (Таблица Animal), id (Таблица Hunting), id (Таблица House), id (Таблица AccountingEvents), id (Таблица User), id (Таблица Gender). Остальные ID-ключи являются внешними ключами.

Для организации более эффективной обработки данных применяется нормализация. Таблицы моей БД находятся в 3НФ:

* БД находится в форме -1НФ потому, что
  + Таблица не имеет повторяющихся записей;
  + Каждый атрибут отношения хранит одно-единственное значение и не является списком, ни множеством значений;
  + Таблица не имеет повторяющихся групп полей.
    - Вторая нормальная форма(2НФ):
      * Устранены атрибуты, зависящие только от части уникального (первичного) идентификатора, т.е. ID.
        + Третья нормальная форма(3НФ):

Отсутствуют атрибуты, зависящие от атрибутов, не входящих в уникальный (первичный) ключ.

На основе реляционной модели базы данных мною в MS SQL была построена следующая база данных. (Рис.7) В ней первичными ключами являются поле таблиц «User» – «id», «AccountingEvents» - «id», «Hunting» - «id», «Animal» - «id». Связь таблиц «Gender» и «User» по полю «GenderId» один ко многим (1:М). Таблицы «Role» и «User» связаны по полю «RoleId» типом связи 1:М. Таблицы «User» и «AccountingEvents» связаны по полю «ClientId» типом связи 1:М. Таблицы «Hunting» и «AccountingEvents» связаны по полю «Hunt» типом связи 1:М. Таблицы «Animal» и «Hunting» связаны по полю «AnimalId» типом связи 1:М. Таблицы «House» и «Hunting» связаны по полю «HouseId» типом связи 1:М. Таблицы «Animal» и «Difficulty» связаны по полю «DifficultyId» типом связи 1:М. Все связи между таблицами базы данных типа один ко многим.



Рис.7 Схема базы данных

Таблица «Gender»(Пол)

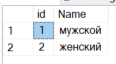


Рис.8 Таблица Gender

Таблица «Role»(Роль)



Рис.9 Таблица Role

Таблица «User»(Пользователи)



Рис.10 Таблица User

Таблица «AccountingEvents»(Учет мероприятий)

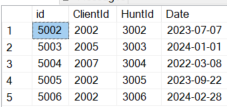


Рис.11 Таблица AccountingEvents

Таблица «Difficulty»(Сложность охоты)

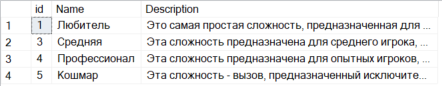


Рис.12 Таблица Difficulty

Таблица « Animal»(Животные)

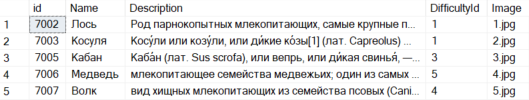


Рис.13 Таблица Animal

Таблица «Hunting»(Охоты)

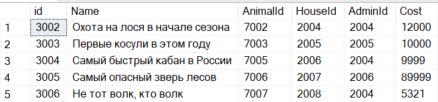


Рис.14 Таблица Hunting

Таблица «House»(Дома)

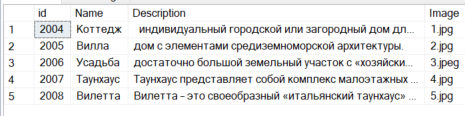


Рис.15 Таблица House

**3.6 Разработка интерфейса ПС**

В приложении предусмотрена авторизация, так как оно многопользовательское. Для запуска приложения достаточно щелкнуть дважды левой кнопкой мыши по ярлыку.

После старта приложения пользователь приступит к работе с главной формой программы, на которой и будут отображаться все события в базе данных. «Huntd.01» обладает удобным интерфейсом, так что даже неопытному в обращение с пользовательскими компьютерными программами пользователю не составит труда быстро привыкнуть к работе с данным программным средством.

На рабочем экране расположены несколько кнопок для комфортной работы с приложением.

**Охотничьи мероприятия**. Основная форма приложения, в которой в виде плитки отображаются мероприятия. Компоненты формы: ListBox – для отображения мероприятий, Combobox – выпадающие списки, TextBox и Button. (Рис.16)



TextBox

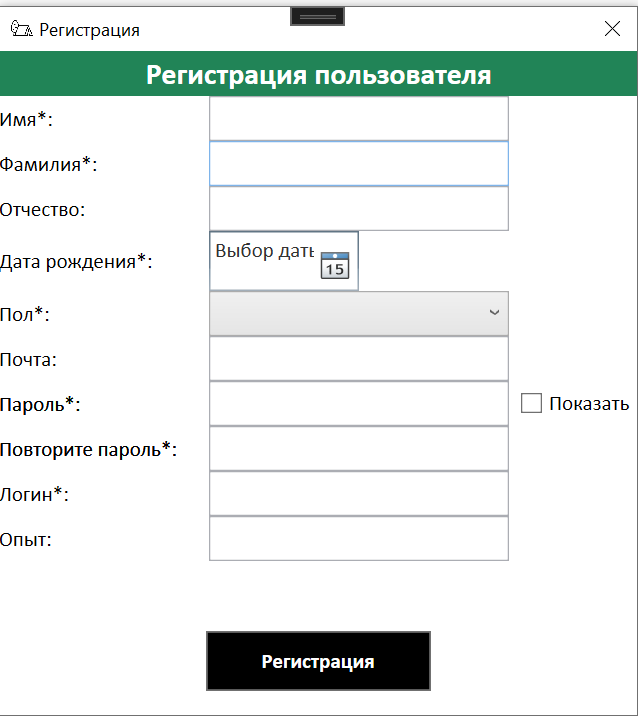
Button

ComboBox

ListBox

Рис.16 Охотничьи мероприятия

**Окно регистрации**. Компоненты Страницы: Combobox – выпадающие списки, DatePicker – для выбора даты, CkeckBox – для проверки видимости пароля, TextBox и Button для произведения регистрации. (Рис.17)



CheckBox

ComboBox

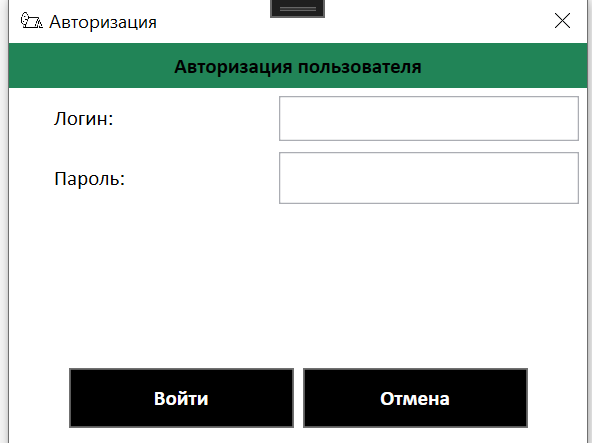
Button

TextBox

DatePicker

Рис.17 Регистрация

**Окно Авторизации**. Интерфейс данной формы состоит из компонентов: TextBox – поля ввода, Image – для отображения капчи и Button. Также есть возможность открытия дополнительных окон. (Рис.18)

****

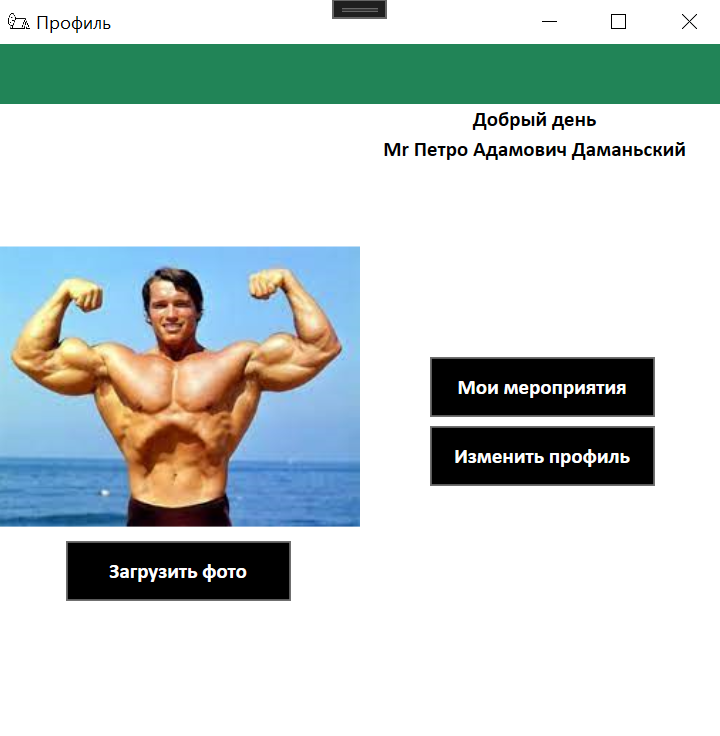
Image

TextBox

Button

Рис.18 Страница Авторизации

**Окно профиля пользователя**. Интерфейс данного окна состоит из компонентов: Image, и Button.(Рис.19)

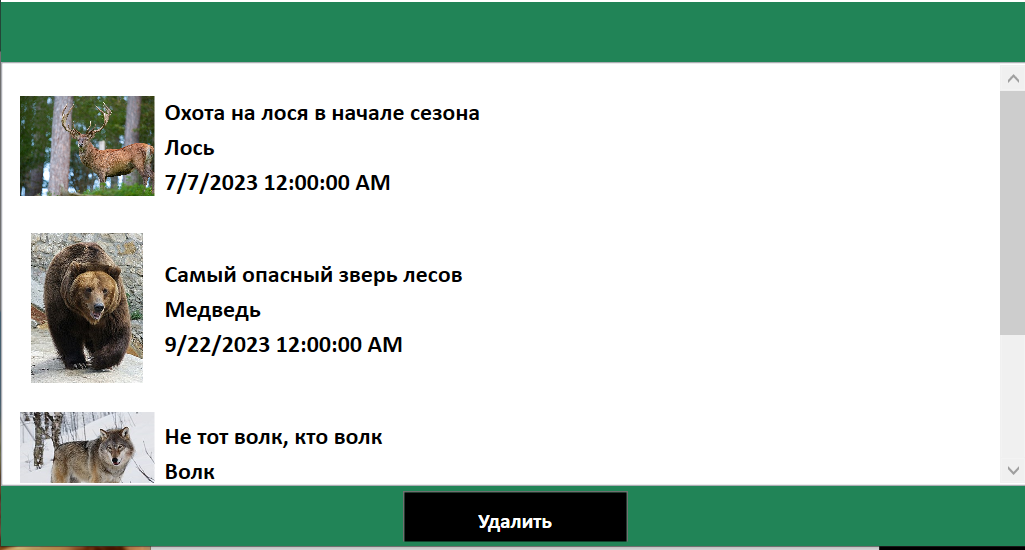


Button

Image

Рис.19 Окно профиля пользователя

**Окно мероприятий пользователя**. Интерфейс данного окна состоит из компонентов: ListBox – для отображения мероприятий пользователя, и Button.(Рис.20)



ListBox

Button

Рис.20 Окно мероприятий пользователя

**Окно изменения профиля**. Интерфейс данной формы является интерфейсом регистрации с уже заполненными полями.(Рис.21)

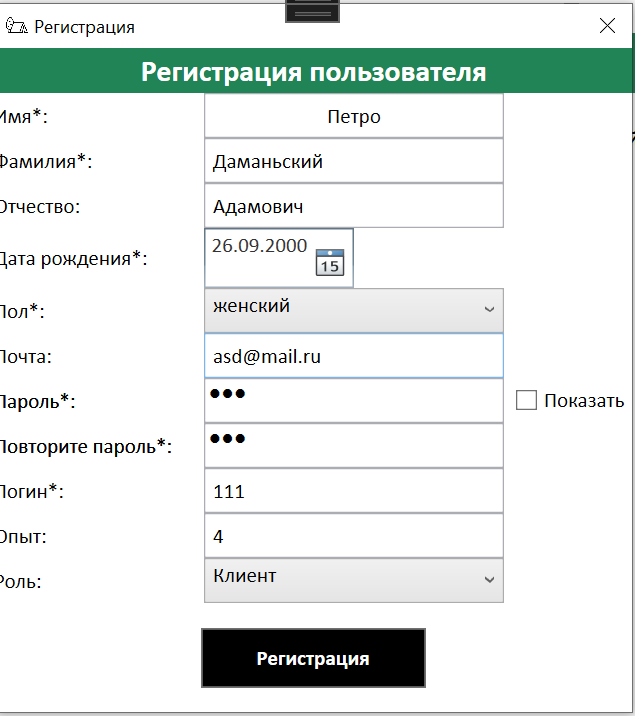
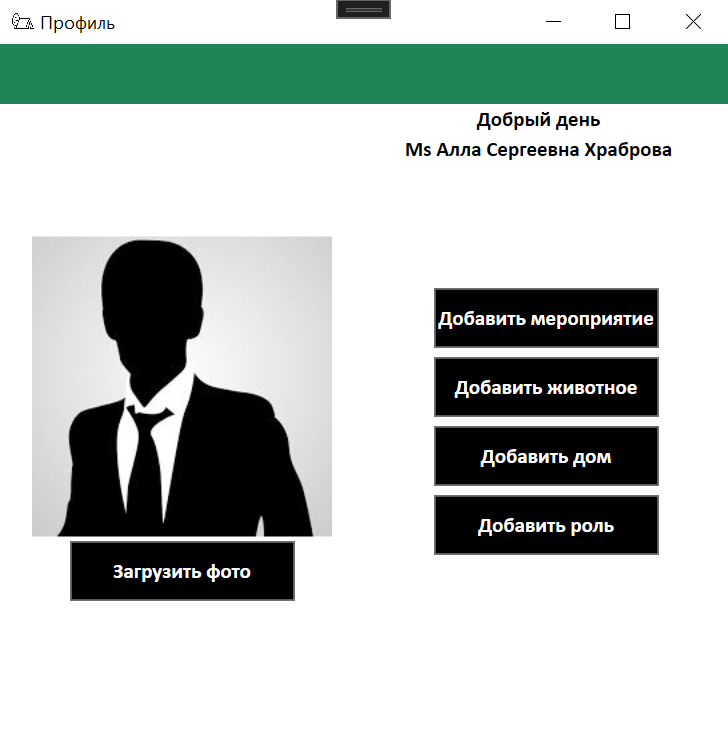


Рис.21 Окно изменения профиля

**Окно профиля модератора**. Интерфейс данного окна состоит из компонентов: Image, и Button.(Рис.22)

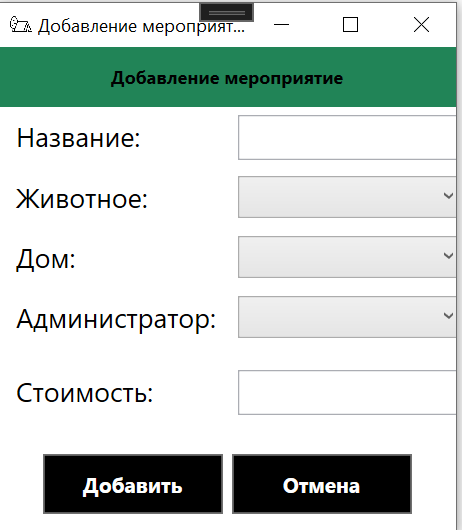


Button

Image

Рис.22 окно профиля модератора

**Окно добавления мероприятия**. Интерфейс данного окна состоит из компонентов:TextBox, ComboBox, и Button.(Рис.23)



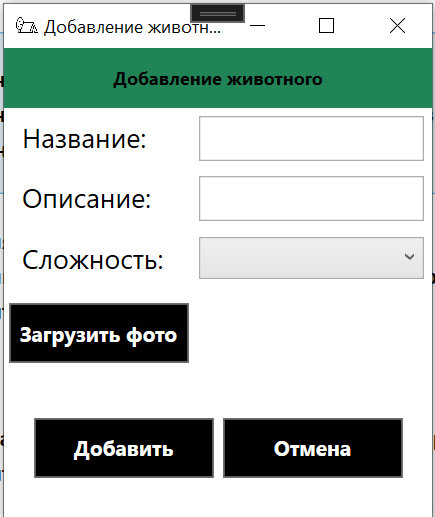
Button

ComboBox

TextBox

Рис.23 окно добавления мероприятия

**Окно добавления животного**. Интерфейс данного окна состоит из компонентов:TextBox, ComboBox, Image и Button.(Рис.24)

****

Button

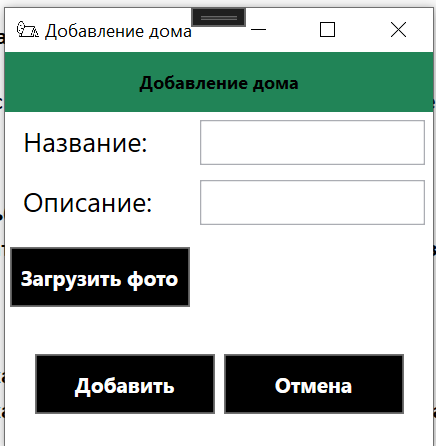
Image

ComboBox

TextBox

Рис.24 окно добавления животного

**Окно добавления дома**. Интерфейс данного окна состоит из компонентов:TextBox, Image и Button.(Рис.25)

****

Button

Image

TextBox

Рис.25 окно добавления дома

**3.7 Описание структуры выходной информации**

Выходными данными являются списки мероприятий отфильтрованные по разным критериям.

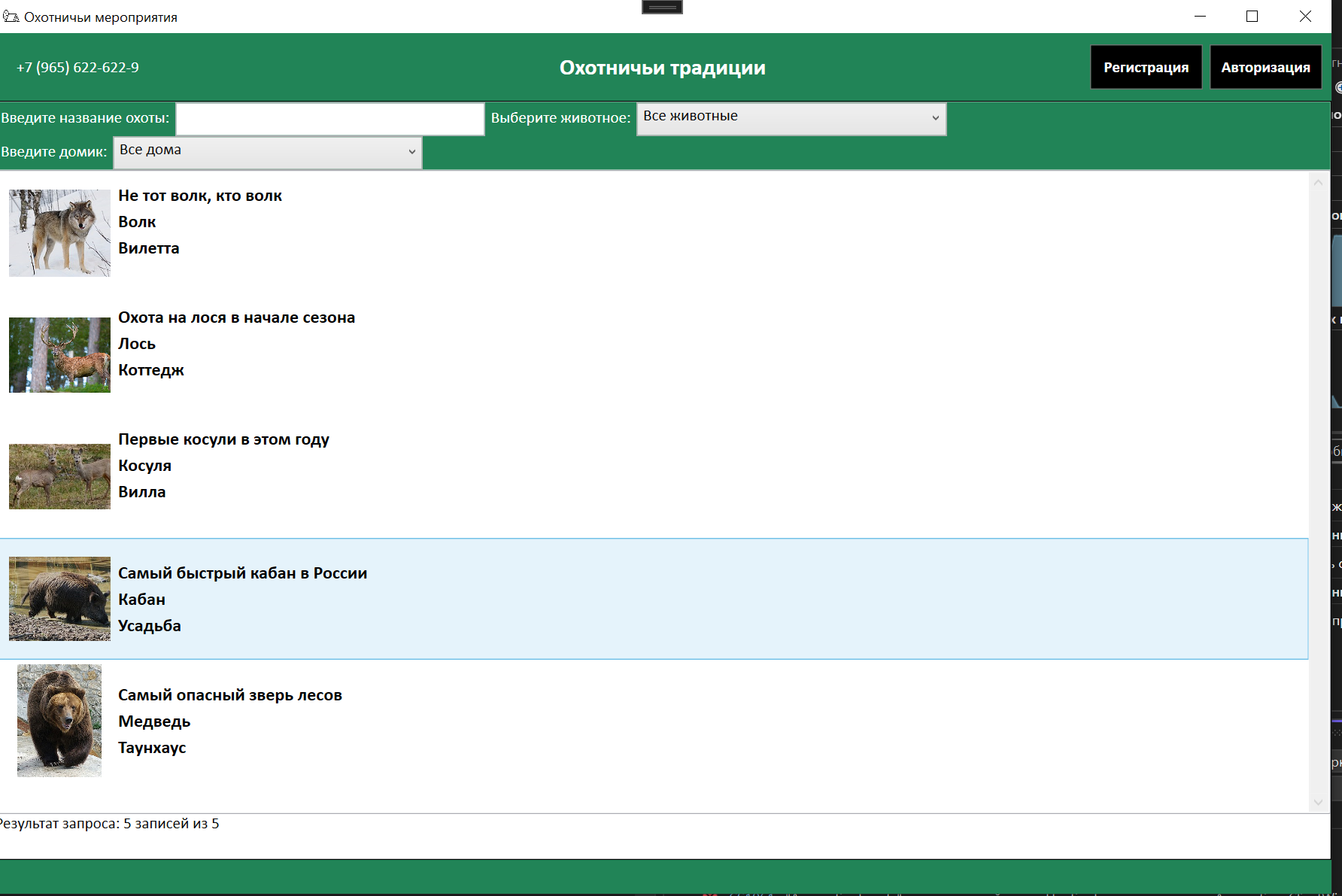


Рис.26 Списки мероприятий

Выходными данными являются списки животных.

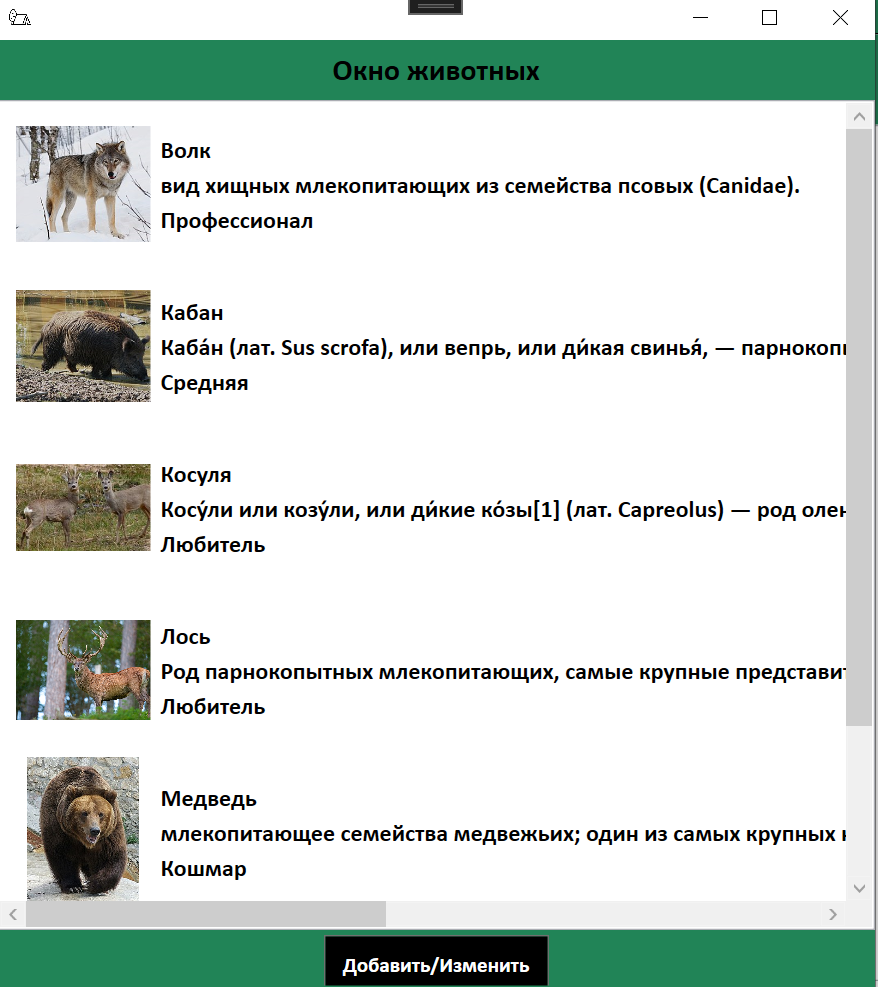


Рис.27 Списки животных

Выходными данными являются списки домов.

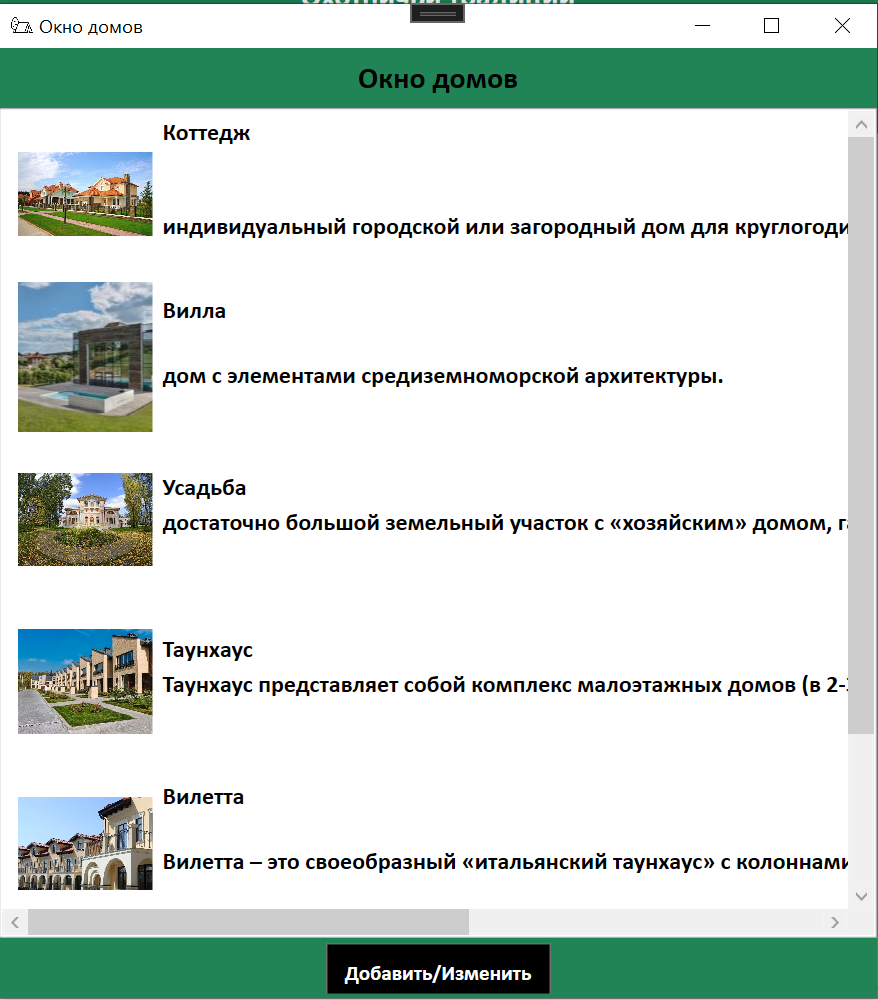


Рис.28 Списки домов

Выходными данными являются списки домов.

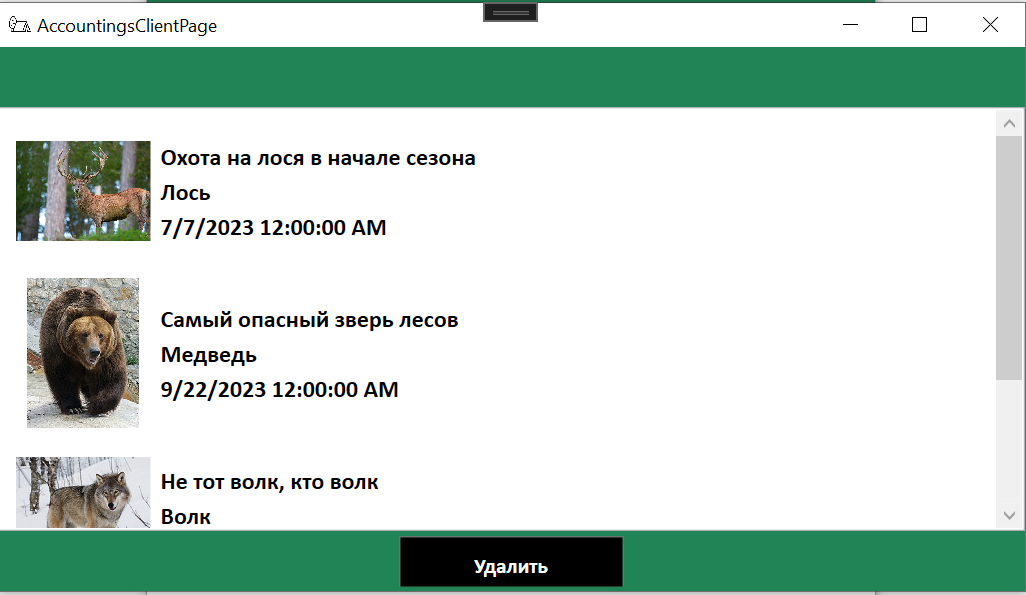


Рис.29 Списки мероприятий клиента

**4. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ**

**4.1 Назначение программного средства**

Назначением программного средства является автоматизация учета мероприятий.

**4.2 Условия выполнения программного средства**

Условия, при соблюдении которых возможно выполнение работы с программным продуктом:

-Компьютер с установленной на нём «Hunt.01».

Подготовительные действия:

-На иконке «Hunt.01» рабочего стола произвести двойной щелчок левой кнопкой мыши.

**4.3 Эксплуатация программного средства**

Этот раздел содержит обзор приложений «Hunt.01» и ее основных параметров.

***Активация продукта***

Активировать копию программы Hunt.01 можно сразу после её установки, когда появится окно подтверждения подлинности Вашей версии. Ключ активации расположен внутри упаковки программного продукта. Для успешного выполнения активации ключ активации необходимо вводить именно в том виде, в котором он предоставлен.

Вводные сведения об интерфейсе пользователя

После старта приложения откроется главное окно приложения – Каталог мероприятий в списочном интерфейсе. На форме есть кнопка для регистрации и авторизации. Есть поля для поиска мероприятия по назвванию и выпадающие списки для фильтрации. (Рис.30)

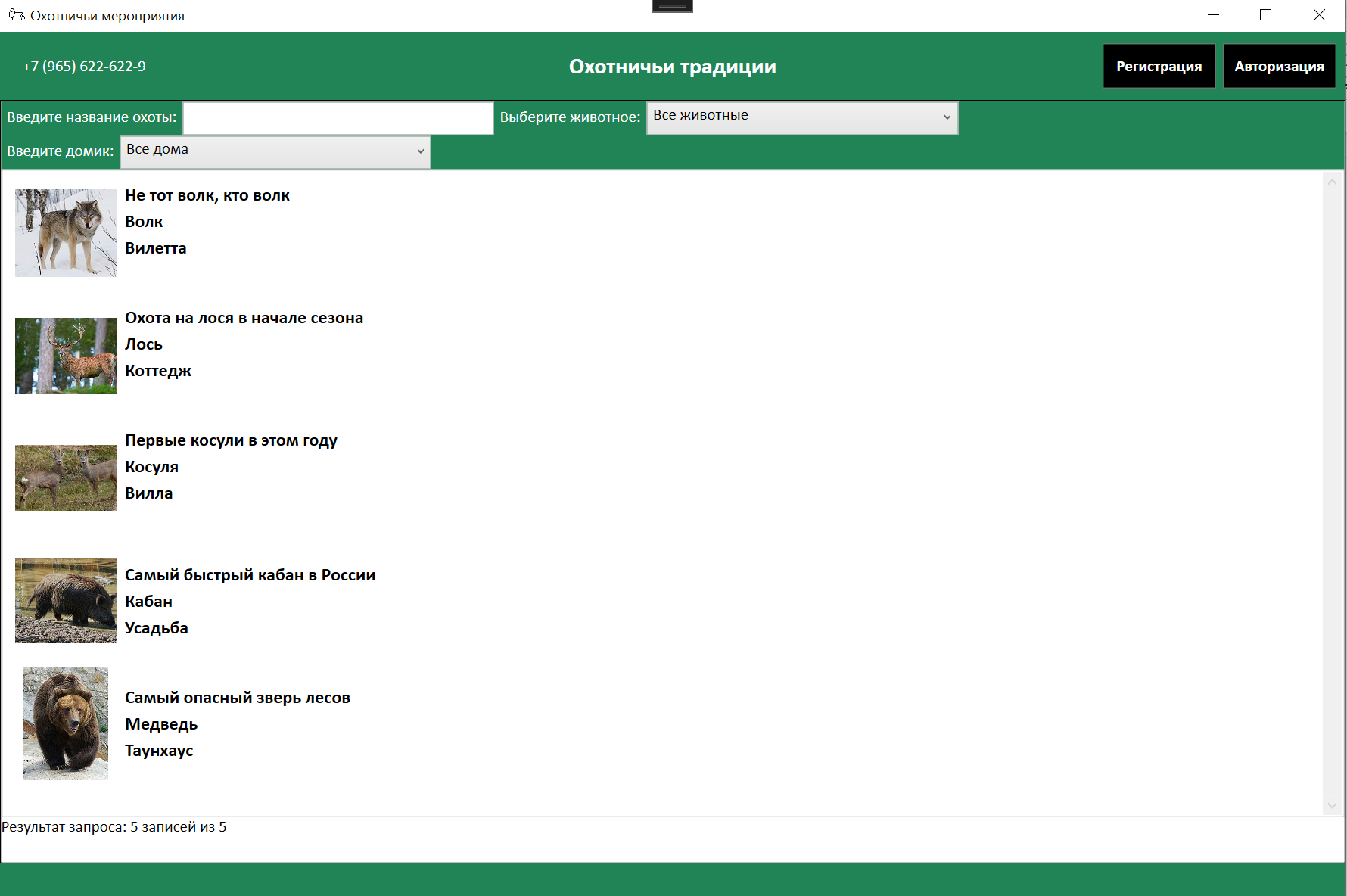


Рис.30 Окно Мероприятия

Кликнув по кнопке «Выход» программа потребует подтверждение. (Рис.31)

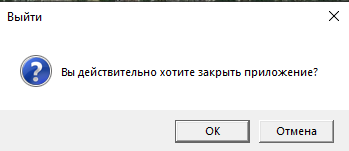


Рис.31 Сообщение программы

Ниже описаны пункты меню управления и информации.

«Окно Авторизация»: эта форма предназначена для прохождения авторизации и предоставления уровня доступа. (Рис.32)

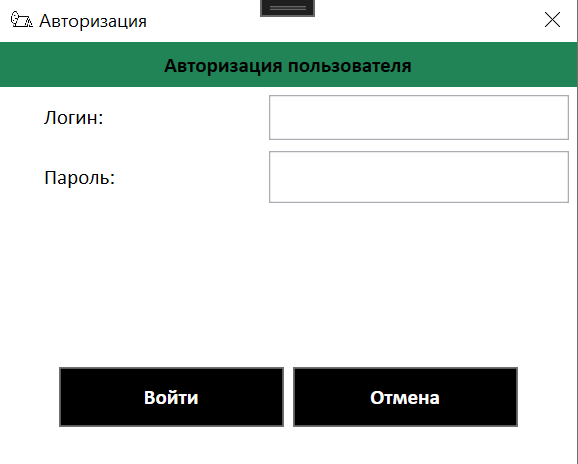


Рис.32 Окно Авторизация

«Окно Регистрация»: эта форма предназначена для прохождения регистрации и предоставления уровня доступа. (Рис.33)

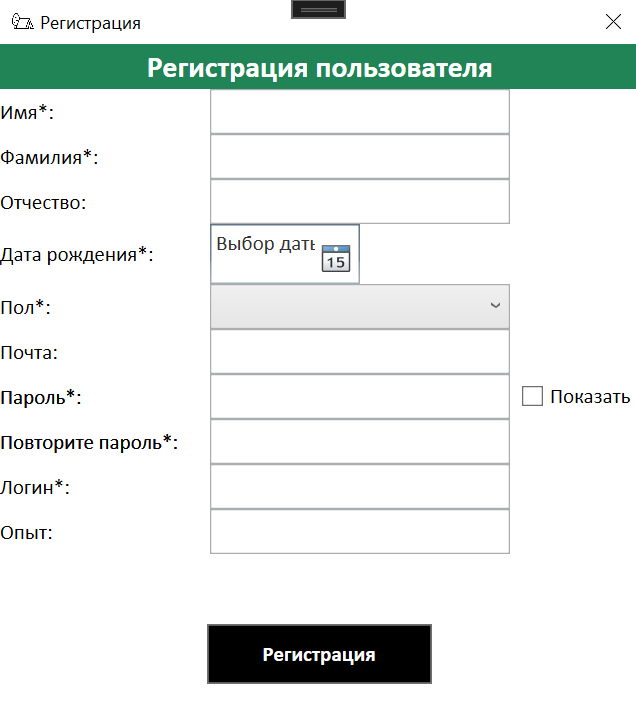


Рис.33 Страница Регистрация

«Окно Профиль»: эта форма предназначена для просмотра профиля текущего пользователя. (Рис.34)

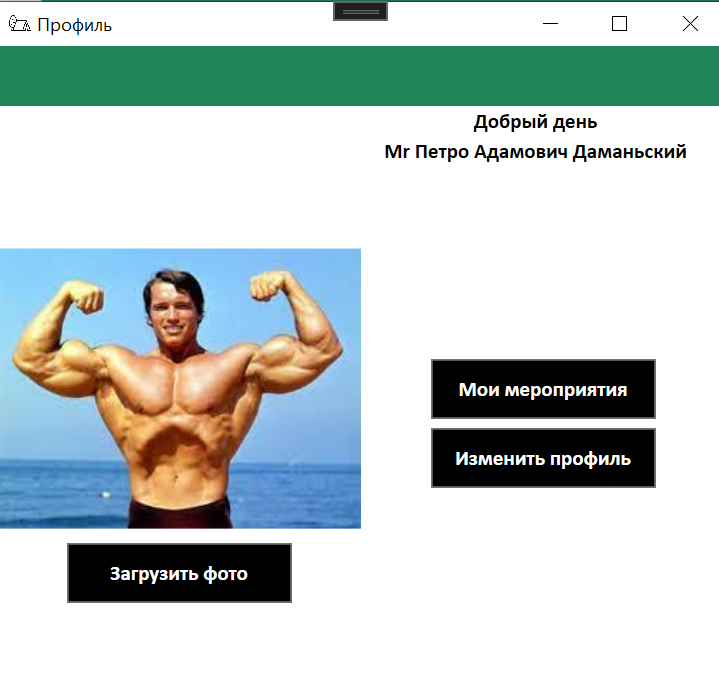


Рис.34 Окно Профиль

«Окно Добавление/изменение мероприятий»: эта форма предназначена для добавления и изменения мероприятия администратором. (Рис.35)

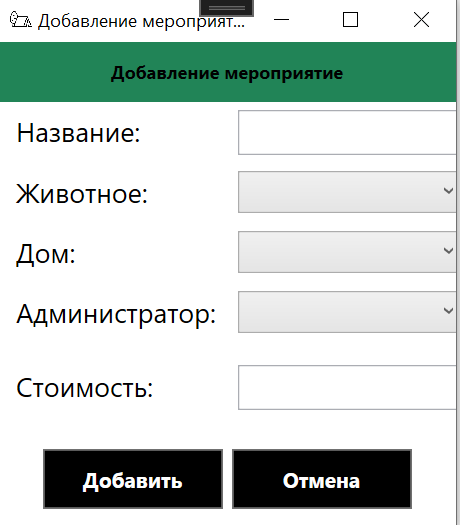


Рис.35 Окно Добавления/изменения мероприятия

«Окно Добавление/изменение животного»: эта форма предназначена для добавления и изменения животных администратором. (Рис.36)

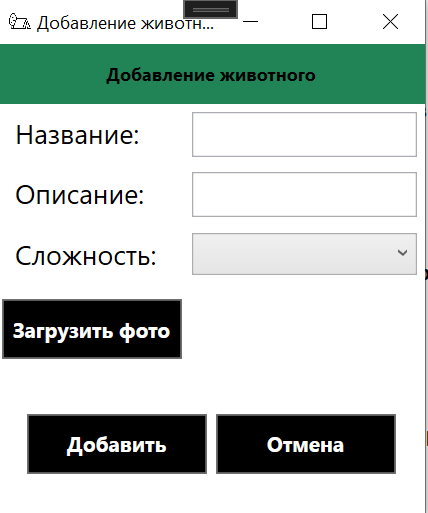


Рис.36 Окно Добавления/изменения животного

«Окно Добавление/изменение дома»: эта форма предназначена для добавления и изменения домов администратором. (Рис.37)

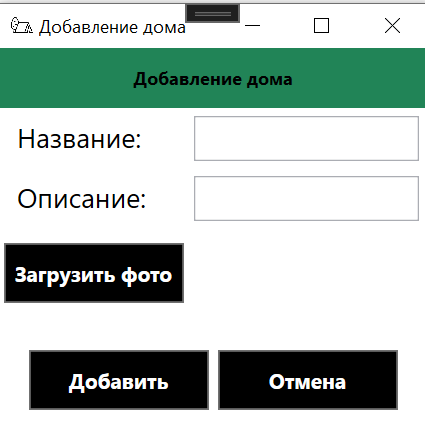


Рис.37 Окно Добавления/изменения дома

«Окно Животных»: эта форма предназначена для просмотра всех животных. (Рис.38)

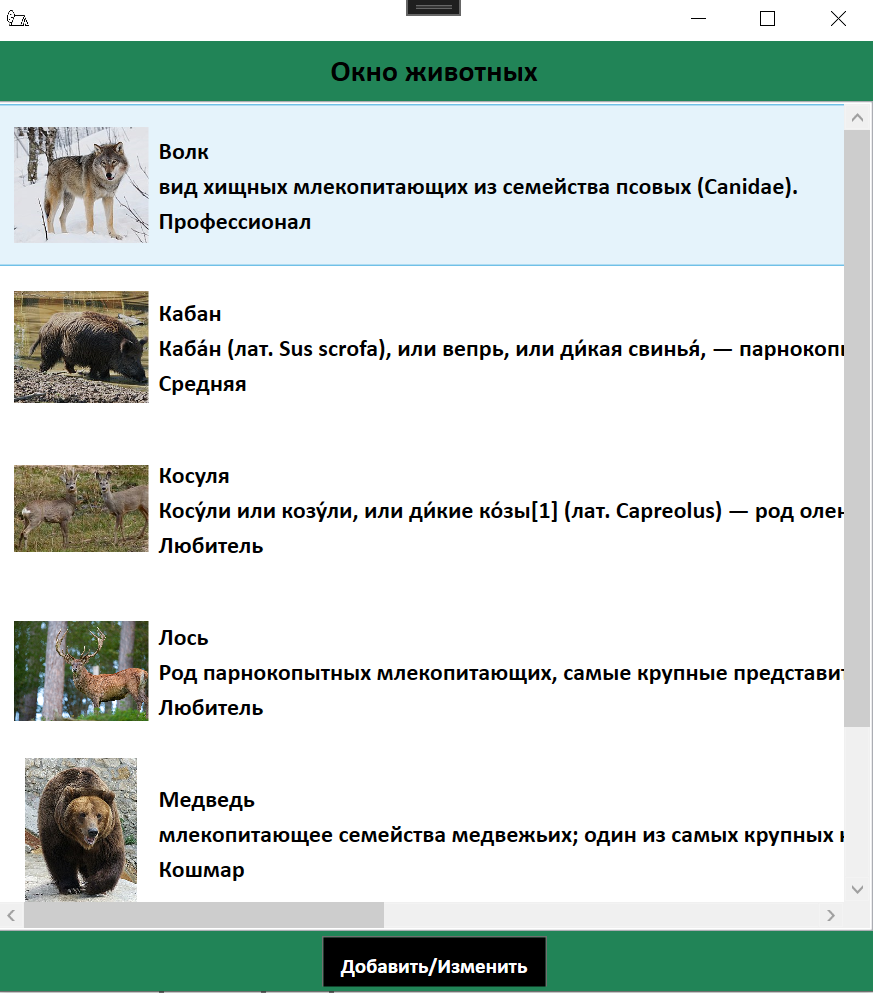


Рис.38 Окно Животных

«Окно Домов»: эта форма предназначена для просмотра всех домов. (Рис.39)

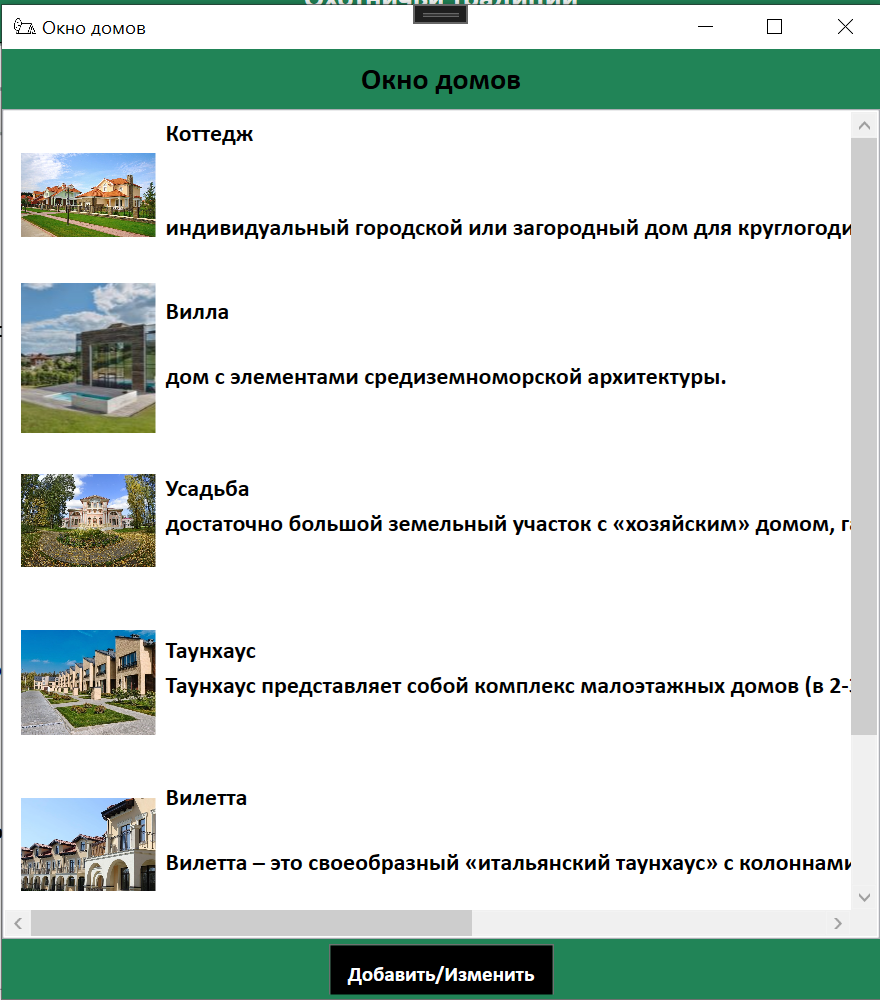


Рис.39 Окно Домов

Работа с Hunt.01

После запуска программы Вы перейдете на главную форму программы. В формах «Мероприятия», «Животные», «Дома», «Регистрация», «Авторизация» , «Добавление и удаление» заполняем все необходимые для работы пункты.

После этого кликаем по кнопке «ОК».

Если Вы желаете удалить любую введенную ранее информацию, Вам необходимо кликнуть по нужной кнопке.

Для того чтобы изменить запись, введите в редактируемые поля данные и нажмите кнопку сохранить.

После того, как Вы закончите сеанс работы с программой, если Вы хотите выйти из Hunt.01 кликните на соответствующей кнопке «Выход».

Основные кнопки Hunt.01

Кнопки управления «Добавить запись», «Удалить запись», «Сохранить» расположены на каждой конкретной форме в базе данных.

Действия этих кнопок приведены в таблице 1:

Таблица-1 Действие кнопок

|  |  |
| --- | --- |
| Добавить запись | Прежде чем добавить запись в таблицу необходимо нажать на данную кнопку. Поля формы будут очищены, пользователю требуется ввести необходимые данные. Если пользователь нажмет кнопку «Сохранить данные», то введенная информация добавится в БД. Если нажать кнопку «Отменить», то введенная информация пользователем удалится. |
| Удалить запись/отменить | Кнопка «Удалить запись» позволяет отменить действия при вводе, либо удаляет конкретную запись. |
| Сохранить данные/обновить запись | Кнопка «Сохранить данные» добавляет запись введенную пользователем в БД, либо обновляет существующую. |
| Редактировать | Открывает форму в режиме редактирования информации |
|  | «Поиск» позволяет найти нужную информацию. |

**4.4 Сообщения пользователю**

Для уверенной и комфортной работы пользователю важно понимать поведение программы, а программе, в свою очередь, поведение пользователя. Для информирования и уточнения намерений пользователей в программах используются сообщения.

При попытке удалить запись, пользователю выводится сообщение с запросом на подтверждение или отмену действия. (Рис.40)

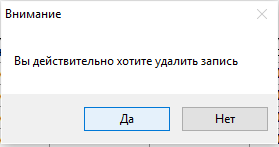


Рис.40 Сообщение Подтверждение удаления

После добавления новой записи, пользователю выводится сообщение. (Рис.41)

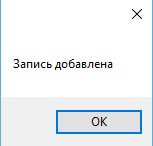


Рис.41 Сообщение Запись добавлена

После обновления записи, пользователю выводится сообщение. (Рис.42)

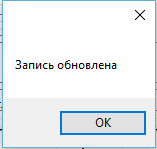


Рис.42 Сообщение Запись обновлена

При попытке удалить связанные записи, программа выводит пользователю сообщение о невозможности действия. (Рис.43)

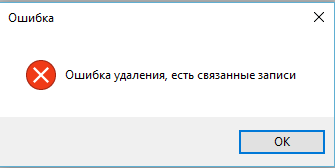


Рис.43 Сообщение Ошибка удаления

**5. РАЗРАБОТКА ТЕСТОВОГО НАБОРА**

**5.1 Обоснование необходимого количества тестов**

Для разработанного алгоритма был построен граф.

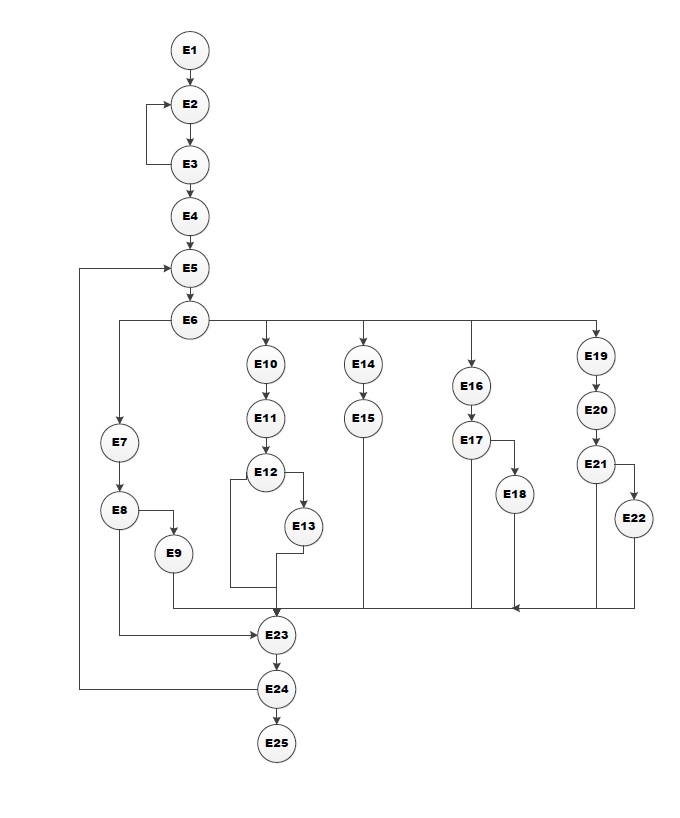
****

Рис.44 Потоковый граф

Необходимое количество тестов определяется по формуле V(G) = E-N+2, где E – это число ребер, а N–это число вершин данного графа. Согласно представленному графу E=34, N=25. Далее по формуле рассчитаем необходимое количество тестовых сценариев (тест-кейсов).

V(G)=34-25+2=11.

Для данного программного средства необходимо разработать не менее 11 тестовых сценариев.

Чтобы определить количество тестовых прогонов для программного продукта Hunt.01 необходимо выделить требования. Из требований к приложению выделим поддержку 2-х операционных систем с одним основным языком локализации и выполнения 6-ти основных функций. Кроме того, приложение позволяет выполнять 5 функций по взаимодействию с системой (запуск приложения, выход из приложения, сохранение результатов в файл, работа с буфером и т.п.).

Таким образом, полное покрытие требований задаёт набор из 2\*1\*(6+5) =22 тестовых прогонов.

**5.2 Описание тестовых пакетов**

Таблица 2 – Тестирование корректного добавления информации в Охота

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного добавления информации в таблицу Охота | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Охота | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку добавить  Заполните форму  Name = Охота на кабанчика  AnimalId = 1  HouseId = 1  AdminId = 1  Cost = 12000 | | Данные успешно введены |  |
| Нажмите кнопку «ОК» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Охота открыта | | Отображается форма Охота |  |

Таблица 3 – Тестирование некорректного добавления информации в таблицу Охота

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного добавления информации в таблицу Охота | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Охота | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку добавить  Заполните форму  Name = Охота на кабанчика  AnimalId =  HouseId = 1  AdminId = 2  Cost = 12000 | | Данные успешно введены |  |
| Нажмите кнопку «ОК» | | Кнопка «ОК» заблокирована, так как поле пустое | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Охота открыта | | Отображается форма Охота |  |

Таблица 4 – Тестирование корректного редактирования информации в таблице Охота

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного редактирования информации в таблице Охота | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Охота | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные,  Name = Охота на кабанчика  AnimalId = 1  HouseId = 1  AdminId = 2  Cost = 12000 | | Данные успешно введены |  |
| Нажмите кнопку «ОК» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Охота  открыта | | Отображается форма Охота |  |

Таблица 5 – Тестирование некорректного редактирования информации в таблице Охота

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного редактирования информации в таблице Мероприятия | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Охота | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные,  Name = Охота на кабанчика  AnimalId = 1  HouseId = 1  AdminId = 22  Cost = 12000 | | Данные успешно введены |  |
| Нажмите кнопку «ОК» | | Кнопка «ОК» заблокирована, поля пустые | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Охота  открыта | | Отображается форма Охота |  |

Таблица 6 – Тестирование корректного добавления информации в таблицу Дом

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного добавления информации в таблицу Дом | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Дом | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку добавить  Заполните форму  Name = Вилла  Description = дом с элементами средиземноморской архитектуры  Image = загружено | | Данные успешно введены |  |
| Нажмите кнопку «ОК» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Дом открыта | | Отображается форма Дом |  |

Таблица 7 – Тестирование некорректного добавления информации в таблицу Дом

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного добавления информации в таблицу Дом | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Дом | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку добавить  Заполните форму  Name =  Description =  Image = загружено | | Данные успешно введены |  |
| Нажмите кнопку «ОК» | | Кнопка «ОК» заблокирована, поля пустые | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Дом  открыта | | Отображается форма Дом |  |

Таблица 8 – Тестирование корректного редактирования информации в таблице Дом

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного редактирования информации в таблице Дом | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Дом | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные,  Name = Вилла  Description = дом с элементами средиземноморской архитектуры  Image = загружено | | Данные успешно введены |  |
| Нажмите кнопку «ОК» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Дом открыта | | Отображается форма Дом |  |

Таблица 9 – Тестирование некорректного редактирования информации в таблице Дом

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного редактирования информации в таблице Дом | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Дом | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные,  Name =  Description =  Image = загружено | | Данные успешно введены |  |
| Нажмите кнопку «ОК» | | Кнопка «ОК» заблокирована, поля пустые | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Дом открыта | | Отображается форма Дом |  |

Таблица 10 – Тестирование корректного добавления информации в Животные

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного добавления информации в таблицу Животные | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Животные | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку добавить  Заполните форму  Name = Кабан  Description = Быстро бегает  DifficultyId = 3  Image = Загружено | | Данные успешно введены |  |
| Нажмите кнопку «ОК» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Животные открыта | | Отображается форма Животные |  |

Таблица 11 – Тестирование некорректного добавления информации в таблицу Животные

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного добавления информации в таблицу Животные | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Животные | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку добавить  Заполните форму  Name =  Description =  DifficultyId = 3  Image = Загружено | | Данные успешно введены |  |
| Нажмите кнопку «ОК» | | Кнопка «ОК» заблокирована, так как поле пустое | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Животные открыта | | Отображается форма Животные |  |

Таблица 12 – Тестирование корректного редактирования информации в таблице Животные

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного редактирования информации в таблице Животные | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Животные | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные,  Name = Кабан  Description = Быстро бегает  DifficultyId = 3  Image = Загружено | | Данные успешно введены |  |
| Нажмите кнопку «ОК» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Животные открыта | | Отображается форма Животные |  |

Таблица 13 – Тестирование некорректного редактирования информации в таблице Животные

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного редактирования информации в таблице Животные | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Животные | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные,  Name =  Description = 1  DifficultyId = 2  Image = Загружено | | Данные успешно введены |  |
| Нажмите кнопку «ОК» | | Кнопка «ОК» заблокирована, поля пустые | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Животные открыта | | Отображается форма Животные |  |

Таблица 14 – Тестирование корректного добавления информации в Пользователи

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного добавления информации в таблицу Пользователи | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Пользователи | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку добавить  Заполните форму  Name = Александр  Surname = Семенов  Patronymic = Сергеевич  Birthday = 19.04.2003  RoleId = Клиент  GenderId = Мужской  Email = sasha@mail.ru  Password = sashaSemen1.  Login = sashaSemen1.  Experience = 4  Image = Загружено | | Данные успешно введены |  |
| Нажмите кнопку «ОК» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Пользователи открыта | | Отображается форма Пользователи |  |

Таблица 15 – Тестирование некорректного добавления информации в таблицу Пользователи

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного добавления информации в таблицу Пользователи | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Пользователи | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Нажмите кнопку добавить  Заполните форму  Name =  Surname =  Patronymic =  Birthday =  RoleId =  GenderId =  Email =  Password =  Login =  Experience =  Image = Загружено | | Данные успешно введены |  |
| Нажмите кнопку «ОК» | | Кнопка «ОК» заблокирована, так как поле пустое | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Пользователи открыта | | Отображается форма Пользователи |  |

Таблица 16 – Тестирование корректного редактирования информации в таблице Пользователи

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование корректного редактирования информации в таблице Пользователи | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Пользователи | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные,  Name = Павел  Surname = Иванов  Patronymic = Сергеевич  Birthday = 19.04.2003  RoleId = Клиент  GenderId = Мужской  Email = sasha@mail.ru  Password = sashaSemen1.  Login = sashaSemen1.  Experience = 4  Image = Загружено | | Данные успешно введены |  |
| Нажмите кнопку «ОК» | | Данные успешно сохранены | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Пользователи открыта | | Отображается форма Пользователи |  |

Таблица 17 – Тестирование некорректного редактирования информации в таблице Пользователи

|  |  |  |  |
| --- | --- | --- | --- |
| **Название:** | Тестирование некорректного редактирования информации в таблице Пользователи | | |
| **Функция:** | Циклы | | |
| **Действие** | | **Ожидаемый результат** | **Результат теста:**   * пройден * провален * заблокирован |
| **Предусловие:** | |  |  |
| Открываем форму Пользователи | | Форма открыта |  |
| **Шаги теста:** | |  |  |
| Выберите запись для редактирования и измените данные,  Name =  Surname =  Patronymic =  Birthday = 19.04.2003  RoleId = Клиент  GenderId = Мужской  Email = sasha@mail.ru  Password = sashaSemen1.  Login = sashaSemen1.  Experience = 4  Image = Загружено | | Данные успешно введены |  |
| Нажмите кнопку «ОК» | | Кнопка «ОК» заблокирована, поля пустые | **Тест пройден** |
| **Постусловие:** | |  |  |
| Форма Пользователи открыта | | Отображается форма Пользователи |  |

**6. ЗАКЛЮЧЕНИЕ**

**6.1 Оценка качества программного средства с помощью метрик**

Оценка качества программного средства проводится исходя из стандарта оценки качества ISО 9126.

На компьютере следующей конфигурации:

Установленная операционная система – Windows 7 Ultimate 32бит.

Процессор: Intel Pentium G2030 3.0 GHz.

ОЗУ: 2 Gb

Таблица 22 – Требования к количественным характеристикам качества программного средства

|  |  |  |  |
| --- | --- | --- | --- |
| **Характеристики качества** | **Мера** | **Требуемое значение** | **Реальное значение** |
| **Надежность** |  | | |
| *Завершенность:*  *-* наработка на отказ при отсутствии рестарта; | Часы | 10 | 10 |
| *Устойчивость:*  - относительные ресурсы на обеспечение надежности и рестарта. | % | 0,5 | 0,5 |
| *Восстанавливаемость:*  -длительность восстановления. | Минуты | 2 | 2 |
| *Доступность-готовность:*  -относительное время работоспособного функционирования. | Вероятность | 0,998 | 0,998 |
| **Эффективность** |  | | |
| *Временная эффективность:*  *-*время отклика - получение результатов на типовое задание;  - пропускная способность число типовых заданий, исполняемых в единицу времени. | Секунды  Число в минуту | 3  60 | 3  60 |
| *Используемость ресурсов:*  *-* относительная величина использования ресурсов ЭВМ при нормальном функционировании программного средства. |  | 0,001 | 0,001 |

Таблица 23 – Требования к качественным характеристикам программного средства.

|  |  |  |  |
| --- | --- | --- | --- |
| **Характеристики качества** | **Мера** | **Требуемое значение** | **Реальное значение** |
| **Практичность** |  | | |
| *Простота использования:*  **-**среднее время ввода заданий;  -среднее время отклика на задание. | Секунды  Секунды | 10  3 | 10  3 |
| *Изучаемость :*  - трудоемкость изучения применения ПС;  -продолжительность изучения;  -объем эксплуатационной документации; | Чел.-часы  Часы  Страницы | 5  2  15 | 5  2  15 |
| **Сопровождаемость** |  | | |
| *Изменяемость:*  -трудоемкость подготовки изменений;  -длительность подготовки изменений. | Чел.-часы  Часы | 5  5 | 5  5 |
| *Тестируемость:*  *-трудоемкость тестирования изменений;*  *- длительность тестирования изменений.* | Чел.-часы  Часы | 3  3 | 3  3 |
| **Мобильность** |  | | |
| *Адаптируемость:*  - трудоемкость адаптации;  -длительность адаптации. | Чел.-часы  Часы | 0,2  1 | 0,2  1 |
| *Простота установки:*  -трудоемкость инсталляции;  -длительность инсталляции. | Чел.-часы  Часы | 0,2  1 | 0,2  1 |
| *Замещаемость:*  -трудоемкость замены компонентов;  - длительность замены компонентов. | Чел.-часы  Часы | 0,2  0,2 | 0,2  0,2 |

Таблица 24 – Сложность

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Тип объекта | Количество | Вес | | | Итого |
|  |  | Простой | Средний | Сложный |  |
| Экран | 9 | x1 | x2 | **x3** | =27 |
| Отчет | 2 | x2 | **x5** | x8 | =10 |

Таблица 25 – Сложность экрана

|  |  |  |  |
| --- | --- | --- | --- |
| Экраны | Количество Категорииских (клт) таблиц данных | | |
| Количество представлений | Всего <4 | **Всего <8** | Всего >8 |
|  | (<2 срв, <3 клт) | **(2-3 срв, 3-5 клт)** | (>3 срв, >5 клт) |
| <3 | Простой | Простой | Средний |
| 3-7 | Простой | Средний | Сложный |
| **>8** | Средний | **Сложный** | Сложный |

Таблица 26 – Сложность отчета

|  |  |  |  |
| --- | --- | --- | --- |
| Отчеты | Количество Категорииских (клт) таблиц данных | | |
| Количество представлений | Всего <4 | Всего <8 | Всего >8 |
|  | (<2 срв, <3 клт) | **(2-3 срв, 3-5 клт)** | (>3 срв, >5 клт) |
| 0 или 1 | Простой | Простой | Средний |
| **2 или 3** | Простой | **Средний** | Сложный |
| >4 | Средний | Сложный | Сложный |

Производительность = 4 тысяч строк кода/2 месяца= 2 тысяч строк кода/месяц

Качество = 2 ошибки /4 тысяч строк кода=0,5 ошибок/тысяч строк кода

**7. СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Г. Шилдт C#: Учебный курс. - Спб.: Питер, 2021. - 512с

2. Либерти, Д. Программирование на Си Шарп. - М.: Символ-плюс, 2016. - 684 с.

3. Ватсон К. Си Шарп/К.Ватсон.-М.:Лори, 2021.-862 с

4. Балена Ф., Димауро Д. Современная практика программирования на Visual Basic и Visual C#.-М.:Русская редакция, 2021,604с.

5. Енин А., Енин Н. Локальная СУБД своими руками. Учимся на примерах.-М.:СОЛОН-ПРЕСС, 2021,464с.,СД

6. Лабор В.В. Си Шарп: СоЗдание приложений для Windows/-Мн.: Харвест, 2021-384с.

7. Фролов А.В., Фролов Г.В. Визуальное проектирование приложений C#.-М.:КУДИЦ-ОБРАЗ, 2021-512с.

8. ГОСТ 19.201-78.ЕСПД. Техническое задание. Требования к содержанию и оформлению

9. ГОСТ 19.102-77.ЕСПД. Стадии разработки

10. ГОСТ 19.505-79.ЕСПД. Руководство оператора. Требования к содержанию и оформлению

11. ГОСТ 19.002-80.ЕСПД. Схемы алгоритмов и программ. Правила выполнения

12. ГОСТ 19.404-79.ЕСПД. Пояснительная записка. Требования к содержанию и оформлению

13. Владимир Биллинг. Основы программирования на С#. 2021г.

[Электронный ресурс http://mexalib.com/view/42671]

14. Никита Культин. Основы программирования в MicrosoftVisualC# 2020. 2020г.

[Электронный ресурс http://www.litres.ru/nikita-kultin/osnovy-programmirovaniya-v-microsoft-visual-c-2021-2]

15. Фролов А.В., Фролов Г.В. Визуальное проектирование приложений C#.-М.:КУДИЦ-ОБРАЗ, 2021.

[Электронный ресурс http://www.twirpx.com/file/27852]

16. Йен Гриффитс. Программирование на С#. 2021г.

[Электронный ресурс http://rutracker.org/forum/viewtopic.php?t=4772853]

**Приложение А**

Разработка интегрированного программного модуля «Охотхозяйство ООО Охотничьи традиции»

Текст кода программного средства

460.ЗМК.00002-01 12 01

**Текст кода программного средства**

**Словарь ресурсов**

<Application x:Class="HuntingFarm.App"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:local="clr-namespace:HuntingFarm"

StartupUri="Windows/MainWindow.xaml">

<Application.Resources>

<SolidColorBrush x:Key="main" Color="#ffffff"/>

<SolidColorBrush x:Key="additional" Color="#218457"/>

<SolidColorBrush x:Key="akcent" Color="Black"/>

<Style TargetType="Window" x:Key="base\_window">

<Setter Property="FontFamily" Value="Calibri"/>

<Setter Property="FontSize" Value="14"/>

<Setter Property="MinHeight" Value="600"/>

<Setter Property="MinWidth" Value="800"/>

<Setter Property="Background" Value="{StaticResource main}"/>

</Style>

<!--стиль для страницы-->

<Style TargetType="Page" x:Key="base\_page">

<Setter Property="FontFamily" Value="Calibri"/>

<Setter Property="Background" Value="{StaticResource main}"/>

<Setter Property="FontSize" Value="14"/>

</Style> <!--стиль для полей ввода TextBox--> <Style TargetType="TextBox"> <Setter Property="Height" Value="30"/> <Setter Property="VerticalAlignment" Value="Stretch"/> <Setter Property="VerticalContentAlignment" Value="Center"/> <Setter Property="Padding" Value="3 0"/> <Setter Property="Background" Value="White"/> </Style> <!--стиль для полей кнопок Button--> <Style TargetType="Button"> <Setter Property="Background" Value="{StaticResource akcent}"/> <Setter Property="Foreground" Value="{StaticResource main}"/> <Setter Property="FontSize" Value="14"/> <Setter Property="FontWeight" Value="Bold"/> <Setter Property="Margin" Value="3"/> <Setter Property="Padding" Value="2"/> <Setter Property="Height" Value="40"/> <Setter Property="Width" Value="150"/> </Style> <!--стиль для меток TextBlock --> <Style TargetType="TextBlock" x:Key="base\_textblock"> <Setter Property="Height" Value="30"/> <Setter Property="VerticalAlignment" Value="Stretch"/> <Setter Property="HorizontalAlignment" Value="Stretch"/> <Setter Property="Padding" Value="5"/> <Setter Property="Foreground" Value="{StaticResource main}"/> </Style> <Style TargetType="TextBlock" x:Key="add\_window\_textblock"> <Setter Property="VerticalAlignment" Value="Center"/> <Setter Property="HorizontalAlignment" Value="Right"/> <Setter Property="FontSize" Value="14"/> <Setter Property="FontWeight" Value="Bold"/> <Setter Property="Margin" Value="0 0 10 0"/> </Style> <!--стиль для меток TextBlock --> <Style TargetType="TextBlock" x:Key="item\_textblock"> <Setter Property="TextAlignment" Value="Center"/> <Setter Property="VerticalAlignment" Value="Top"/> <Setter Property="TextWrapping" Value="Wrap"/> <Setter Property="FontSize" Value="14"/> <Setter Property="Padding" Value="3"/> </Style> <Style TargetType="TextBlock" x:Key="textblock\_textbox\_combo"> <Setter Property="Width" Value="120"/> <Setter Property="FontSize" Value="18"/> <Setter Property="Padding" Value="2"/> </Style> <!--стиль для меток ListViewItem --> <Style TargetType="ListBoxItem" x:Key="good\_item"> <Setter Property="Background" Value="{Binding GetColor}" /> <Setter Property="BorderBrush" Value="{StaticResource akcent}"/> <Setter Property="BorderThickness" Value="1"/> <Setter Property="HorizontalContentAlignment" Value="Stretch"/> </Style> </Application.Resources> </Application> **Класс Manager** using System.Windows.Controls; namespace HuntingFarm.Models { public partial class Manager { public static Frame MainFrame {get;set;} public static User CurrentUser { get; set; } } } **Класс Animal.cs** using System.IO; namespace HuntingFarm.Models { public partial class Animal { public string GetAnimalImage { get { if (Image is null) return Directory.GetCurrentDirectory() + @"\Images\default.png"; return Directory.GetCurrentDirectory() + @"\Images\Animals\" + Image.Trim(); } } [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")] public Animal() { this.Huntings = new HashSet<Hunting>(); } public int id { get; set; } public string Name { get; set; } public string Description { get; set; } public int DifficultyId { get; set; } public string Image { get; set; } public virtual Difficulty Difficulty { get; set; } [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")] public virtual ICollection<Hunting> Huntings { get; set; } } } **Класс House.cs** using System; using System.Collections.Generic; using System.IO; namespace HuntingFarm.Models { public partial class House { [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")] public House() { this.Huntings = new HashSet<Hunting>(); } public int id { get; set; } public string Name { get; set; } public string Description { get; set; } public string Image { get; set; } [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")] public virtual ICollection<Hunting> Huntings {

get; set; } public string GetHouseImage { get { if (Image is null) return Directory.GetCurrentDirectory() + @"\Images\default.png"; return Directory.GetCurrentDirectory() + @"\Images\Houses\" + Image.Trim(); } } } } **Класс HuntEntities.cs** using System.Data.Entity; namespace HuntingFarm.Models { public partial class HuntEntities : DbContext { private static HuntEntities \_context; public static HuntEntities GetContext() { if (\_context == null) \_context = new HuntEntities(); return \_context; } } } Класс User.cs using System; using System.Collections.Generic; using System.IO; namespace HuntingFarm.Models { [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")] public User() { this.AccountingEvents = new HashSet<AccountingEvent>(); this.Huntings = new HashSet<Hunting>(); } public int id { get; set; } public string Name { get; set; } public string Surname { get; set; } public string Patronymic { get; set; } public System.DateTime Birthday { get; set; } public int RoleId { get; set; } public int GenderId { get; set; } public string Email { get; set; } public string Password { get; set; } public string Login { get; set; } public Nullable<int> Experience { get; set; } public string Image { get; set; } [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")] public virtual ICollection<AccountingEvent> AccountingEvents { get; set; } public virtual Gender Gender { get; set; } [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")] public virtual ICollection<Hunting> Huntings { get; set; } public virtual Role Role { get; set; } public partial class User { public string GetPhoto { get { if (Image == null) return Directory.GetCurrentDirectory() + @"\Images\default.png"; if (RoleId == 1) return Directory.GetCurrentDirectory() + @"\Images\Clients\" + Image.Trim(); if (RoleId == 2) return Directory.GetCurrentDirectory() + @"\Images\Administrators\" + Image.Trim(); return Directory.GetCurrentDirectory() + @"\Images\default.png"; } } } } **Класс MackeCaptcha** using System; using System.Drawing; using System.IO; using System.Windows.Media.Imaging; namespace HuntingFarm.Models { public class MakeCaptcha { private static BitmapImage BitmapToImageSource(Bitmap bitmap) { using (MemoryStream memory = new MemoryStream()) { bitmap.Save(memory, System.Drawing.Imaging.ImageFormat.Bmp); memory.Position = 0; BitmapImage bitmapimage = new BitmapImage(); bitmapimage.BeginInit(); bitmapimage.StreamSource = memory; bitmapimage.CacheOption = BitmapCacheOption.OnLoad; bitmapimage.EndInit(); return bitmapimage; } } public static (BitmapImage image, string captcha) CreateImage(int width, int height, int symbolCount) { Random rnd = new Random(); Bitmap result = new Bitmap(width, height); //Добавим различные цвета Brush[] colors = { Brushes.Black, Brushes.Red, Brushes.RoyalBlue, Brushes.Green }; //Укажем где рисовать Graphics g = Graphics.FromImage((System.Drawing.Image)result); //Пусть фон картинки будет серым g.Clear(Color.Gray); //Сгенерируем текст char symbol; // нбор допустимых символов string alphabet = "1234567890QWERTYUIOPASDFGHJKLZXCVBNM"; // текст капчи string captcha = ""; // размер поля для одного символа int h = width / symbolCount; for (int i = 0; i < symbolCount; ++i) { // генерируем размер буквы int size = rnd.Next(20, h); // выбираем любой символ из допустимого набора symbol = alphabet[rnd.Next(alphabet.Length)]; // текст капчи captcha += symbol; // генерируем позиции рисования символа int Ypos = rnd.Next(height - size - 20); int Xpos = rnd.Next(i \* h, (i + 1) \* h - size); //Нарисуем сгенерированный символ g.DrawString(symbol.ToString(), new Font("Arial", size), colors[rnd.Next(colors.Length)], new PointF(Xpos, Ypos)); //Добавим немного помех ///////Линии из углов //Устанавливаем цвет линии Pen skyBluePen = new Pen(colors[rnd.Next(colors.Length)]); // Устанавливаем толщину линии skyBluePen.Width = rnd.Next(2, 5); g.DrawLine(skyBluePen, new Point(Xpos + 5, Ypos + 5), new Point(Xpos + size + 5, Ypos + size + 5)); } ////Белые точки for (int i = 0; i < width; ++i) for (int j = 0; j < height; ++j) if (rnd.Next() % 20 == 0) result.SetPixel(i, j, Color.White); return (BitmapToImageSource(result), captcha); } } } }**Форма Главная Код интерфейса** <Window x:Class="HuntingFarm.MainWindow" xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation" xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" xmlns:d="http://schemas.microsoft.com/expression/blend/2008" xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006" xmlns:local="clr-namespace:HuntingFarm" mc:Ignorable="d" Title="{Binding ElementName=MainFrame, Path=Content.Title}" WindowStartupLocation="CenterScreen" Style="{StaticResource base\_window}" Height="800" Width="1200" Closing="Window\_Closing" Closed="Window\_Closed"> <Grid> <Grid.RowDefinitions> <RowDefinition Height="60"/> <RowDefinition Height="\*"/> <RowDefinition Height="30"/> </Grid.RowDefinitions> <Grid Background="{StaticResource additional}"/> <TextBlock Grid.Row="0" Text="Охотничьи традиции" HorizontalAlignment="Center" VerticalAlignment="Center" Foreground="{StaticResource main}" FontSize="20" FontWeight="Bold" /> <Frame x:Name="MainFrame" Grid.Row="1" BorderBrush="{StaticResource akcent}" NavigationUIVisibility="Hidden" BorderThickness="1"/> <TextBlock Text="+7 (965) 622-622-9" Foreground="{StaticResource main}" Grid.Row="0" HorizontalAlignment="Left" VerticalAlignment="Center" Margin="20 0 0 0"/> <StackPanel Orientation="Horizontal" HorizontalAlignment="Right" Margin="5"> <Button x:Name="btnRegistration" Content="Регистрация" Grid.Row="0" Click="BtnRegistration\_Click" Width="100"/> <Button x:Name="btnAutorization" Content="Авторизация" Grid.Row="0" Click="BtnAutorization\_Click" Width="100"/> </StackPanel> <TextBlock x:Name="TextBlockName" Width="Auto" Height="40" HorizontalAlignment="Right" VerticalAlignment="Top"/> <Grid Grid.Row="2" Background="{StaticResource additional}"/> </Grid> </Window>  **Код программы** using System; using System.Windows; using HuntingFarm.Pages; using HuntingFarm.Models; using HuntingFarm.Windows; namespace HuntingFarm { /// <summary> /// Логика взаимодействия для MainWindow.xaml /// </summary> public partial class MainWindow : Window { public MainWindow() { InitializeComponent(); LoadData(); } void LoadData() { MainFrame.Navigate(new HuntingEventsPage()); Manager.MainFrame = MainFrame; } private void Window\_Closing(object sender, System.ComponentModel.CancelEventArgs e) { MessageBoxResult x = MessageBox.Show("Вы действительно хотите выйти?", "Выйти", MessageBoxButton.OKCancel, MessageBoxImage.Question); if (x == MessageBoxResult.Cancel) e.Cancel = true; } private void Window\_Closed(object sender, EventArgs e) { Application.Current.Shutdown(); } private void BtnRegistration\_Click(object sender, RoutedEventArgs e) { try { RegistrationWindow registrationWindow = new RegistrationWindow(); if (registrationWindow.ShowDialog() == true) { btnRegistration.Visibility = Visibility.Hidden; btnAutorization.Content = "Профиль"; btnAutorization.Click += HandleClickProfile; btnAutorization.Click -= BtnAutorization\_Click; MainFrame.Navigate(new HuntingEventsPage()); Manager.MainFrame = MainFrame; } } catch { MessageBox.Show("Ошибка"); } } private void BtnAutorization\_Click(object sender, RoutedEventArgs e) { try { AutorizationWindow window = new AutorizationWindow(); if (window.ShowDialog() == true) { btnRegistration.Content = "Выйти"; btnRegistration.Click += HandleClickSignOut; btnRegistration.Click -= BtnRegistration\_Click; btnAutorization.Content = "Профиль"; btnAutorization.Click += HandleClickProfile; btnAutorization.Click -= BtnAutorization\_Click; MainFrame.Navigate(new HuntingEventsPage()); Manager.MainFrame = MainFrame; } } catch { MessageBox.Show("Ошибка"); } } private void HandleClickProfile(object sender, RoutedEventArgs args) { ProfileWindow profileWindow = new ProfileWindow(); profileWindow.Show(); } private void HandleClickSignOut(object sender, RoutedEventArgs args) { Manager.CurrentUser = null; btnAutorization.Content = "Авторизация"; btnAutorization.Click -= HandleClickProfile; btnAutorization.Click += BtnAutorization\_Click; btnRegistration.Content = "Регистрация"; btnRegistration.Click -= HandleClickSignOut; btnRegistration.Click += BtnRegistration\_Click; MainFrame.Navigate(new HuntingEventsPage()); Manager.MainFrame = MainFrame; } } }  **Страница Мероприятия Код интерфейса** <Page x:Class="HuntingFarm.Pages.HuntingEventsPage" xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation" xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006" xmlns:d="http://schemas.microsoft.com/expression/blend/2008" xmlns:local="clr-namespace:HuntingFarm.Pages" mc:Ignorable="d" d:DesignHeight="450" d:DesignWidth="800" Title="Охотничьи мероприятия" IsVisibleChanged="Page\_IsVisibleChanged" Style="{StaticResource base\_page}"> <Grid> <Grid.RowDefinitions> <RowDefinition Height="auto"/> <RowDefinition Height="\*"/> <RowDefinition Height="40"/> </Grid.RowDefinitions> <WrapPanel Grid.Row="0" HorizontalAlignment="Stretch" Background="{StaticResource additional}"> <StackPanel Orientation="Horizontal"> <TextBlock Text="Введите название охоты:" Style="{StaticResource base\_textblock}"/> <TextBox Width="275" x:Name="TbSearch" TextChanged="TbSearch\_TextChanged"/> </StackPanel> <StackPanel Orientation="Horizontal"> <TextBlock Text="Выберите животное:" Style="{StaticResource base\_textblock}"/> <ComboBox Width="275" x:Name="ComboAnimal" DisplayMemberPath="Name" SelectionChanged="ComboAnimal\_SelectionChanged"/> </StackPanel> <StackPanel Orientation="Horizontal"> <TextBlock Text="Введите домик:" Style="{StaticResource base\_textblock}"/> <ComboBox Width="275" x:Name="ComboHouse" DisplayMemberPath="Name" SelectionChanged="ComboHouse\_SelectionChanged"/> </StackPanel> </WrapPanel> <TextBlock Grid.Row="2" x:Name="TextBlockInfo" Width="300" HorizontalAlignment="Left"/> <ListBox x:Name="ListBoxHuntings" ScrollViewer.VerticalScrollBarVisibility="Visible" SelectedValuePath="id" Grid.Row="1" Grid.Column="0"> <ListBox.ItemTemplate> <DataTemplate> <Grid HorizontalAlignment="Stretch" UseLayoutRounding="True"> <Grid.ColumnDefinitions> <ColumnDefinition Width="100"/> <ColumnDefinition Width="\*"/> </Grid.ColumnDefinitions> <Image Width="100" Height="100" Grid.Column="0" Stretch="Uniform" Margin="5 2 5 2" Source="{Binding Path=Animal.GetAnimalImage}"/> <StackPanel Grid.Column="1" VerticalAlignment="Center" Width="800"> <TextBlock Margin="2" FontSize="16" FontWeight="Bold" Text="{Binding Path=Name}" Width="auto" HorizontalAlignment="Left"/> <TextBlock Margin="2" FontSize="16" FontWeight="Bold" Text="{Binding Path=Animal.Name}" Width="auto" HorizontalAlignment="Left"/> <TextBlock Margin="2" FontSize="16" FontWeight="Bold" Text="{Binding Path=House.Name}" Width="auto" HorizontalAlignment="Left"/> </StackPanel> </Grid> </DataTemplate> </ListBox.ItemTemplate> </ListBox> <Button x:Name="btnSignUpEvent" Grid.Row="2" Content="Записаться" Visibility="Hidden" Click="btnSignUpEvent\_Click"/> </Grid> </Page>  **Код программный**  using System; using System.Collections.Generic; using System.Linq; using System.Windows; using System.Windows.Controls; using HuntingFarm.Models; using HuntingFarm.Windows; namespace HuntingFarm.Pages { /// <summary> /// Логика взаимодействия для HuntingEventsPage.xaml /// </summary> public partial class HuntingEventsPage : Page { int \_itemcount = 0; public HuntingEventsPage() { InitializeComponent(); LoadAndInitData(); } void LoadAndInitData() { ListBoxHuntings.ItemsSource = HuntEntities.GetContext().Huntings.OrderBy(p => p.Name).ToList(); \_itemcount = ListBoxHuntings.Items.Count; List<Animal> animals = HuntEntities.GetContext().Animals.OrderBy(p => p.Name).ToList(); animals.Insert(0, new Animal { Name = "Все животные", Description = "Любое", }); List<House> houses = HuntEntities.GetContext().Houses.OrderBy(p => p.Name).ToList(); houses.Insert(0, new House { Name = "Все дома", Description = "Любое", }); ComboHouse.ItemsSource = houses; ComboHouse.SelectedIndex = 0; ComboAnimal.ItemsSource = animals; ComboAnimal.SelectedIndex = 0; TextBlockInfo.Text = $"Результат запроса: {\_itemcount} записей из {\_itemcount}"; } private void Page\_IsVisibleChanged(object sender, DependencyPropertyChangedEventArgs e) { if (Visibility == Visibility.Visible) { HuntEntities.GetContext().ChangeTracker.Entries().ToList().ForEach(p => p.Reload()); ListBoxHuntings.ItemsSource = HuntEntities.GetContext().Huntings.OrderBy(p => p.Name).ToList(); if (Manager.CurrentUser != null && Manager.CurrentUser.RoleId == 1) btnSignUpEvent.Visibility = Visibility.Visible; } } private void TbSearch\_TextChanged(object sender, TextChangedEventArgs e) { Update(); } private void Update() { var currentHuntings = HuntEntities.GetContext().Huntings.OrderBy(p => p.Name).ToList(); if (ComboAnimal.SelectedIndex > 0) currentHuntings = currentHuntings.Where(p => p.AnimalId == (ComboAnimal.SelectedItem as Animal).id).ToList(); if (ComboHouse.SelectedIndex > 0) currentHuntings = currentHuntings.Where(p => p.HouseId == (ComboHouse.SelectedItem as House).id).ToList(); currentHuntings = currentHuntings.Where(p => p.Name.ToLower().Contains(TbSearch.Text.ToLower())).ToList(); ListBoxHuntings.ItemsSource = currentHuntings; TextBlockInfo.Text = $"Результат запроса: {currentHuntings.Count()} записей из {\_itemcount}"; } private void ComboAnimal\_SelectionChanged(object sender, SelectionChangedEventArgs e) { Update(); } private void ComboHouse\_SelectionChanged(object sender, SelectionChangedEventArgs e) { Update(); } private void btnSignUpEvent\_Click(object sender, RoutedEventArgs e) { if (ListBoxHuntings.SelectedIndex == -1) { MessageBox.Show("Мероприятие не выбрано"); return; } try { SelectDateWindow selectDateWindow = new SelectDateWindow((ListBoxHuntings.SelectedItem as Hunting).id); selectDateWindow.Show(); } catch (Exception ex) { MessageBox.Show(ex.Message); } } } }  **Окно Пользователей Файл интерфейса** <Window x:Class="HuntingFarm.Windows.UsersWindow" xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation" xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" xmlns:d="http://schemas.microsoft.com/expression/blend/2008" xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006" xmlns:local="clr-namespace:HuntingFarm.Windows" mc:Ignorable="d" Title="Окно пользователей" Style="{StaticResource base\_window}" MinHeight="400" MinWidth="600" Width="600" WindowStartupLocation="CenterScreen"> <Grid> <Grid.RowDefinitions> <RowDefinition Height="40"/> <RowDefinition Height="\*"/> <RowDefinition Height="40"/> </Grid.RowDefinitions> <Grid Background="{StaticResource additional}"/> <TextBlock Text="Окно пользователей" HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="20" FontWeight="Bold"/> <Grid Background="{StaticResource additional}" Grid.Row="2"/> <ListBox x:Name="ListBoxUsers" ScrollViewer.VerticalScrollBarVisibility="Visible" SelectedValuePath="id" Grid.Row="1" Grid.Column="0"> <ListBox.ItemTemplate> <DataTemplate> <Grid HorizontalAlignment="Stretch" UseLayoutRounding="True"> <Grid.ColumnDefinitions> <ColumnDefinition Width="100"/> <ColumnDefinition Width="\*"/> </Grid.ColumnDefinitions> <Image Width="100" Height="100" Grid.Column="0" Stretch="Uniform" Margin="5 2 5 2" Source="{Binding Path=GetPhoto}"/> <StackPanel Grid.Column="1" VerticalAlignment="Center"> <TextBlock Margin="2" FontSize="16" FontWeight="Bold" Text="{Binding Path=Surname}" Width="auto" HorizontalAlignment="Left"/> <TextBlock Margin="2" FontSize="16" FontWeight="Bold" Text="{Binding Path=Name}" Width="auto" HorizontalAlignment="Left"/> <TextBlock Margin="2" FontSize="16" FontWeight="Bold" Text="{Binding Path=Patronymic}" Width="auto" HorizontalAlignment="Left"/> <TextBlock Margin="2" FontSize="16" FontWeight="Bold" Text="{Binding Path=Role.Name}" Width="auto" HorizontalAlignment="Left"/> </StackPanel> </Grid> </DataTemplate> </ListBox.ItemTemplate> </ListBox> <Button x:Name="btnAddOrEditUsers" Content="Добавить/Изменить" Grid.Row="2" Click="btnAddOrEditUsers\_Click"/> </Grid> </Window>  **Файл программного кода** 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; using HuntingFarm.Models; namespace HuntingFarm.Windows { /// <summary> /// Логика взаимодействия для UsersWindow.xaml /// </summary> public partial class UsersWindow : Window { public UsersWindow() { InitializeComponent(); ListBoxUsers.ItemsSource = HuntEntities.GetContext().Users.ToList(); } private void btnAddOrEditUsers\_Click(object sender, RoutedEventArgs e) { if (ListBoxUsers.SelectedItem != null) { RegistrationWindow registrationWindow = new RegistrationWindow(ListBoxUsers.SelectedItem as User); registrationWindow.Owner = this; this.Hide(); registrationWindow.Show(); } else { RegistrationWindow registrationWindow = new RegistrationWindow(); registrationWindow.Show(); } } } }  **Окно Регистрации Файл интерфейса** <Window x:Class="HuntingFarm.Windows.RegistrationWindow" xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation" xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" xmlns:d="http://schemas.microsoft.com/expression/blend/2008" xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006" xmlns:local="clr-namespace:HuntingFarm.Windows" mc:Ignorable="d" Title="Регистрация" Height="480" Width="440" Style="{StaticResource base\_window}" MinHeight="400" MinWidth="400" WindowStartupLocation="CenterScreen" ResizeMode="NoResize"> <Grid> <Grid.RowDefinitions> <RowDefinition Height="30"/> <RowDefinition Height="30"/> <RowDefinition Height="30"/> <RowDefinition Height="30"/> <RowDefinition Height="40"/> <RowDefinition Height="30"/> <RowDefinition Height="30"/> <RowDefinition Height="30"/> <RowDefinition Height="30"/> <RowDefinition Height="30"/> <RowDefinition Height="30"/> <RowDefinition Height="30"/> <RowDefinition Height="\*"/> </Grid.RowDefinitions> <Grid.ColumnDefinitions> <ColumnDefinition /> </Grid.ColumnDefinitions> <Grid Background="{StaticResource additional}"/> <TextBlock Text="Регистрация пользователя" HorizontalAlignment="Center" x:Name="TbTitle" VerticalAlignment="Center" FontWeight="Bold" FontSize="20" Foreground="{StaticResource main}" Margin="82,3,81,3"/> <StackPanel Orientation="Horizontal" Grid.Row="1"> <TextBlock Text="Имя\*:" VerticalAlignment="Center" Width="120"/> <TextBox x:Name="TbName" Width="200" Margin="20 0 0 0" TextAlignment="Center"/> </StackPanel> <StackPanel Orientation="Horizontal" Grid.Row="2"> <TextBlock Text="Фамилия\*:" VerticalAlignment="Center" Width="120"/> <TextBox x:Name="TbSurname" Width="200" Margin="20 0 0 0"/> </StackPanel> <StackPanel Orientation="Horizontal" Grid.Row="3"> <TextBlock Text="Отчество:" VerticalAlignment="Center" Width="120"/> <TextBox x:Name="TbPatronymic" Width="200" Margin="20 0 0 0"/> </StackPanel> <StackPanel Orientation="Horizontal" Grid.Row="4"> <TextBlock Text="Дата рождения\*:" VerticalAlignment="Center" Width="120"/> <DatePicker x:Name="DatePickerBirthday" Width="100" VerticalAlignment="Top" HorizontalAlignment="Left" Margin="20 0 0 0"/> </StackPanel> <StackPanel Orientation="Horizontal" Grid.Row="5"> <TextBlock Text="Пол\*:" VerticalAlignment="Center" Width="120"/> <ComboBox x:Name="ComboGender" Width="200" Margin="20 0 0 0" DisplayMemberPath="Name"/> </StackPanel> <StackPanel Orientation="Horizontal" Grid.Row="6"> <TextBlock Text="Почта:" VerticalAlignment="Center" Width="120"/> <TextBox x:Name="TbEmail" Width="200" Margin="20 0 0 0"/> </StackPanel> <StackPanel Orientation="Horizontal" Grid.Row="7"> <TextBlock Text="Пароль\*:" VerticalAlignment="Center" Width="120"/> <PasswordBox x:Name="PasswordBox" Width="200" Margin="20 0 0 0" /> </StackPanel> <StackPanel Orientation="Horizontal" Grid.Row="7"> <TextBlock Text="Пароль\*:" VerticalAlignment="Center" Width="120"/> <TextBox x:Name="TbPassword" Width="200" Margin="20 0 0 0" Visibility="Collapsed"/> </StackPanel> <StackPanel Orientation="Horizontal" Grid.Row="8"> <TextBlock Text="Повторите пароль\*:" VerticalAlignment="Center" Width="120"/> <PasswordBox x:Name="PasswordBoxAgain" Width="200" Margin="20 0 0 0" /> </StackPanel> <StackPanel Orientation="Horizontal" Grid.Row="8"> <TextBlock Text="Повторите пароль\*:" VerticalAlignment="Center" Width="120"/> <TextBox x:Name="TbPasswordAgain" Width="200" Margin="20 0 0 0" Visibility="Collapsed"/> </StackPanel> <CheckBox x:Name="CheckBoxShowPassword" Grid.Row="7" Grid.Column="1" HorizontalAlignment="Right" Margin="0 0 5 0" Checked="CheckBoxShowPassword\_Checked" Unchecked="CheckBoxShowPassword\_Unchecked" VerticalAlignment="Center" Content="Показать"/> <StackPanel Orientation="Horizontal" Grid.Row="9"> <TextBlock Text="Логин\*:" VerticalAlignment="Center" Width="120"/> <TextBox x:Name="TbLogin" Width="200" Margin="20 0 0 0"/> </StackPanel> <StackPanel Orientation="Horizontal" Grid.Row="10"> <TextBlock Text="Опыт:" VerticalAlignment="Center" Width="120"/> <TextBox x:Name="TbExperience" Width="200" Margin="20 0 0 0"/> </StackPanel> <StackPanel Orientation="Horizontal" Grid.Row="11" Visibility="Collapsed" x:Name="StackPanelRole"> <TextBlock Text="Роль:" VerticalAlignment="Center" Width="120"/> <ComboBox x:Name="ComboRole" Width="200" Margin="20 0 0 0" DisplayMemberPath="Name"/> </StackPanel> <Button x:Name="btnRegister" Click="BtnRegister\_Click" Content="Регистрация" Grid.Row="12" Grid.RowSpan="2"/> </Grid> </Window>  **Файл программного кода** 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; using HuntingFarm.Models; namespace HuntingFarm.Windows { /// <summary> /// Логика взаимодействия для RegistrationWindow.xaml /// </summary> public partial class RegistrationWindow : Window { User \_currentUser = null; public RegistrationWindow() { InitializeComponent(); ComboGender.ItemsSource = HuntEntities.GetContext().Genders.ToList(); } public RegistrationWindow(User user) { InitializeComponent(); \_currentUser = user; StackPanelRole.Visibility = Visibility.Visible; ComboGender.ItemsSource = HuntEntities.GetContext().Genders.ToList(); ComboGender.SelectedIndex = Manager.CurrentUser.GenderId; TbName.Text = user.Name; TbSurname.Text = user.Surname; TbPatronymic.Text = user.Patronymic; TbEmail.Text = user.Email; TbExperience.Text = user.Experience.ToString(); TbLogin.Text = user.Login; PasswordBox.Password = user.Password; PasswordBoxAgain.Password = user.Password; TbPassword.Text = user.Password; TbPasswordAgain.Text = user.Password; DatePickerBirthday.SelectedDate = user.Birthday; ComboRole.ItemsSource = HuntEntities.GetContext().Roles.ToList(); ComboRole.SelectedIndex = Manager.CurrentUser.RoleId - 1; } private void BtnRegister\_Click(object sender, RoutedEventArgs e) { StringBuilder \_errors = CheckFields(); if (\_errors.Length > 0) { MessageBox.Show(\_errors.ToString()); return; } try { string passowrd = (bool)CheckBoxShowPassword.IsChecked ? TbPassword.Text : PasswordBox.Password; if (\_currentUser == null) { User user = new User() { Name = TbName.Text, Surname = TbSurname.Text, Patronymic = TbPatronymic.Text, Birthday = (DateTime)DatePickerBirthday.SelectedDate, RoleId = 1, GenderId = (ComboGender.SelectedItem as Gender).id, Email = TbEmail.Text, Password = passowrd, Login = TbLogin.Text, Experience = int.Parse(TbExperience.Text), Image = null, }; HuntEntities.GetContext().Users.Add(user); } else { User user = HuntEntities.GetContext().Users.First(p => p.id == \_currentUser.id); user.Name = TbName.Text; user.Surname = TbSurname.Text; user.Patronymic = TbPatronymic.Text; user.Birthday = (DateTime)DatePickerBirthday.SelectedDate; user.RoleId = (ComboRole.SelectedItem as Role).id; user.GenderId = (ComboGender.SelectedItem as Gender).id; user.Email = TbEmail.Text; user.Password = passowrd; user.Login = TbLogin.Text; user.Experience = int.Parse(TbExperience.Text); user.Image = \_currentUser.Image; } HuntEntities.GetContext().SaveChanges(); MessageBox.Show("Успешно"); } catch (Exception ex) { MessageBox.Show(ex.Message); } } private bool CheckPassword(string password) { if (password.Length < 6) return false; if (password.ToLower() == password || password.ToUpper() == password) return false; int count = 0, spec = 0; for (int i = 0; i < password.Length;i++) { if (char.IsDigit(password[i])) count++; if (!char.IsLetterOrDigit(password[i])) spec++; } if(count == 0 || spec == 0) return false; return true; } private StringBuilder CheckFields() { StringBuilder sb = new StringBuilder(); User login = null; if (\_currentUser == null) login = HuntEntities.GetContext().Users.FirstOrDefault(p => p.Login == TbLogin.Text); if (string.IsNullOrWhiteSpace(TbSurname.Text)) sb.AppendLine("Введите фамилию"); if (string.IsNullOrWhiteSpace(TbName.Text)) sb.AppendLine("Введите имя"); if (!CheckBirthday(DatePickerBirthday.SelectedDate)) sb.AppendLine("Введите дату, либо вам меньше 18"); if (ComboGender.SelectedIndex == -1) sb.AppendLine("Выберите пол"); if ((string.IsNullOrWhiteSpace(PasswordBox.Password) || string.IsNullOrWhiteSpace(PasswordBoxAgain.Password) && CheckBoxShowPassword.IsChecked == false)) sb.AppendLine("Введите пароль"); else if ((string.IsNullOrWhiteSpace(TbPassword.Text) || string.IsNullOrWhiteSpace(TbPasswordAgain.Text) && CheckBoxShowPassword.IsChecked == true)) sb.AppendLine("Введите пароль"); else { if (CheckBoxShowPassword.IsChecked == true) { if (TbPassword.Text != TbPasswordAgain.Text) sb.AppendLine("Пароли отличаются"); else if (!CheckPassword(TbPassword.Text)) sb.AppendLine("Пароль не соответствует требованиям:\n - не менее 6символов; " + "\n - заглавные и строчные буквы;\n - не менее одного спецсимвола;\n - неменее одной цифры."); } else { if(PasswordBox.Password != PasswordBoxAgain.Password) sb.AppendLine("Пароли отличаются"); else if (!CheckPassword(PasswordBox.Password)) sb.AppendLine("Пароль не соответствует требованиям:\n - не менее 6символов; " + "\n - заглавные и строчные буквы;\n - не менее одного спецсимвола;\n - неменее одной цифры."); } } if (string.IsNullOrWhiteSpace(TbLogin.Text)) sb.AppendLine("Введите логин"); if (login != null) sb.AppendLine("Пользователь с таким логином уже существует"); return sb; } private bool CheckBirthday(DateTime? date) { if (date == null) return false; TimeSpan timeSpan = DateTime.Now - date ?? new TimeSpan(); if (timeSpan.TotalDays / 365 < 18) return false; return true; } private void CheckBoxShowPassword\_Checked(object sender, RoutedEventArgs e) { TbPassword.Text = PasswordBox.Password; PasswordBox.Visibility = Visibility.Collapsed; TbPassword.Visibility = Visibility.Visible; TbPasswordAgain.Text = PasswordBoxAgain.Password; PasswordBoxAgain.Visibility = Visibility.Collapsed; TbPasswordAgain.Visibility = Visibility.Visible; } private void CheckBoxShowPassword\_Unchecked(object sender, RoutedEventArgs e) { PasswordBox.Password = TbPassword.Text; TbPassword.Visibility = Visibility.Collapsed; PasswordBox.Visibility = Visibility.Visible; PasswordBoxAgain.Password = TbPasswordAgain.Text; TbPasswordAgain.Visibility = Visibility.Collapsed; PasswordBoxAgain.Visibility = Visibility.Visible; } } }  **Окно Авторизации Файл интерфейса** <Window x:Class="HuntingFarm.Windows.AutorizationWindow" xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation" xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" xmlns:d="http://schemas.microsoft.com/expression/blend/2008" xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006" xmlns:local="clr-namespace:HuntingFarm.Windows" mc:Ignorable="d" Title="Авторизация" Height="200" Width="400" Style="{StaticResource base\_window}" MinHeight="400" MinWidth="400" ResizeMode="NoResize" WindowStartupLocation="CenterScreen"> <Grid> <Grid.RowDefinitions> <RowDefinition Height="30"/> <RowDefinition Height="40"/> <RowDefinition Height="40"/> <RowDefinition Height="40" x:Name="RowCaptcha1"/> <RowDefinition Height="100" x:Name="RowCaptcha2"/> <RowDefinition Height="\*"/> </Grid.RowDefinitions> <Grid Background="{StaticResource additional}"/> <TextBlock Text="Авторизация пользователя" Grid.Row="0" HorizontalAlignment="Center" VerticalAlignment="Center" FontWeight="Bold"/> <StackPanel Orientation="Horizontal" Grid.Row="1"> <TextBlock Text="Логин:" Width="150" Margin="30 0 0 0" VerticalAlignment="Center"/> <TextBox x:Name="TbLogin" Width="200"/> </StackPanel> <StackPanel Orientation="Horizontal" Grid.Row="2"> <TextBlock Text="Пароль:" Width="150" Margin="30 0 0 0" VerticalAlignment="Center"/> <PasswordBox x:Name="TbPassword" Width="200" Height="35"/> </StackPanel> <StackPanel Orientation="Horizontal" Grid.Row="3"> <TextBlock Text="Введите капчу:" Width="150" VerticalAlignment="Center" Margin="30 0 0 0"/> <TextBox x:Name="TbCaptcha" Width="200"/> </StackPanel> <Image x:Name="ImageCaptcha" Height="100" Width="300" Grid.Row="4"/> <Button Width="24" Height="24" Padding="0" Content="0" VerticalAlignment="Center" HorizontalAlignment="Left" x:Name="btnRenewCaptcha" Click="BtnRenewCaptcha\_Click" Grid.Row="3"/> <TextBlock Style="{StaticResource item\_textblock}" Grid.Row="5" x:Name="TbTime" HorizontalAlignment="Center" Width="326"/> <StackPanel Orientation="Horizontal" Grid.Row="5" HorizontalAlignment="Center"> <Button Content="Войти" x:Name="btnEnter" Click="BtnEnter\_Click"/> <Button Content="Отмена" x:Name="btnCancle" Click="BtnCancle\_Click"/> </StackPanel> </Grid> </Window>  **Файл программного кода** using System; using System.Collections.Generic; using System.Linq; using System.Text; using System.Threading; using System.Windows; using System.Windows.Threading; using HuntingFarm.Models; namespace HuntingFarm.Windows { /// <summary> /// Логика взаимодействия для AutorizationWindow.xaml /// </summary> public partial class AutorizationWindow : Window { bool b = false; int count = 0; DispatcherTimer timer = new DispatcherTimer(); int seconds = 0; string captcha = ""; public AutorizationWindow() { InitializeComponent(); LoadAndInitData(); } void LoadAndInitData() { this.Height = 200; timer.Tick += timer\_tick; RowCaptcha1.Height = new GridLength(0); RowCaptcha2.Height = new GridLength(0); } void timer\_tick(object sender, EventArgs e) { seconds -= 1; TbTime.Text = $"Осталось {seconds} секунд до разблокировки"; if (seconds == 0) { btnEnter.IsEnabled = true; TbTime.Text = ""; timer.Stop(); } } private void BtnEnter\_Click(object sender, RoutedEventArgs e) { try { List<User> users = HuntEntities.GetContext().Users.ToList(); User user = users.FirstOrDefault(p => p.Password == TbPassword.Password && p.Login == TbLogin.Text); if ((user != null && !b) || (user != null && b && TbCaptcha.Text == captcha)){ Manager.CurrentUser = user; DialogResult = true; } else { MessageBox.Show("Неверный логин или пароль"); count++; if (count >= 3) { this.Height = 350; ShowCaptcha(); timer.Interval = TimeSpan.FromSeconds(1); btnEnter.IsEnabled = false; seconds = 10; TbTime.Text = $"Осталось {seconds} до разблокировки"; timer.Start(); RowCaptcha1.Height = new GridLength(34); RowCaptcha2.Height = new GridLength(1, GridUnitType.Star); } } } catch (Exception ex) { MessageBox.Show(ex.Message.ToString()); } } private void BtnCancle\_Click(object sender, RoutedEventArgs e) { DialogResult = false; this.Close(); } private void BtnRenewCaptcha\_Click(object sender, RoutedEventArgs e) { ShowCaptcha(); } void ShowCaptcha () { var tuple = MakeCaptcha.CreateImage(300, 100, 4); ImageCaptcha.Source = tuple.image; captcha = tuple.captcha; } } }  **Окно Домов Файл интерфейса** <Window x:Class="HuntingFarm.Windows.HousesWindow" xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation" xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" xmlns:d="http://schemas.microsoft.com/expression/blend/2008" xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006" xmlns:local="clr-namespace:HuntingFarm.Windows" mc:Ignorable="d" Title="Окно домов" Style="{StaticResource base\_window}" MinHeight="400" MinWidth="600" Width="600" WindowStartupLocation="CenterScreen"> <Grid> <Grid.RowDefinitions> <RowDefinition Height="40"/> <RowDefinition Height="\*"/> <RowDefinition Height="40"/> </Grid.RowDefinitions> <Grid Background="{StaticResource additional}"/> <TextBlock Text="Окно домов" HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="20" FontWeight="Bold"/> <Grid Background="{StaticResource additional}" Grid.Row="2"/> <ListBox x:Name="ListBoxHouses" ScrollViewer.VerticalScrollBarVisibility="Visible" SelectedValuePath="id" Grid.Row="1" Grid.Column="0"> <ListBox.ItemTemplate> <DataTemplate> <Grid HorizontalAlignment="Stretch" UseLayoutRounding="True"> <Grid.ColumnDefinitions> <ColumnDefinition Width="100"/> <ColumnDefinition Width="\*"/> </Grid.ColumnDefinitions> <Image Width="100" Height="100" Grid.Column="0" Stretch="Uniform" Margin="5 2 5 2" Source="{Binding Path=GetHouseImage}"/> <StackPanel Grid.Column="1" VerticalAlignment="Center"> <TextBlock Margin="2" FontSize="16" FontWeight="Bold" Text="{Binding Path=Name}" Width="auto" HorizontalAlignment="Left"/> <TextBlock Margin="2" FontSize="16" FontWeight="Bold" Text="{Binding Path=Description}" Width="auto" HorizontalAlignment="Left"/> </StackPanel> </Grid> </DataTemplate> </ListBox.ItemTemplate> </ListBox> <Button x:Name="btnAddOrEditHouse" Content="Добавить/Изменить" Grid.Row="2" Click="btnAddOrEditHouse\_Click"/> </Grid> </Window>  **Файл программного кода** 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; using HuntingFarm.Models; namespace HuntingFarm.Windows { /// <summary> /// Логика взаимодействия для HousesWindow.xaml /// </summary> public partial class HousesWindow : Window { public HousesWindow() { InitializeComponent(); ListBoxHouses.ItemsSource = HuntEntities.GetContext().Houses.ToList(); } private void btnAddOrEditHouse\_Click(object sender, RoutedEventArgs e) { if (ListBoxHouses.SelectedItem != null) { AddOrEditHouseWindow addOrEditHouse = new AddOrEditHouseWindow(ListBoxHouses.SelectedItem as House); addOrEditHouse.Owner = this; this.Hide(); addOrEditHouse.Show(); } else { AddOrEditHouseWindow addOrEditHouse = new AddOrEditHouseWindow(); addOrEditHouse.Show(); } } } }  **Окно Животных Файл интерфейса** <Window x:Class="HuntingFarm.Windows.AnimalsWindow" xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation" xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" xmlns:d="http://schemas.microsoft.com/expression/blend/2008" xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006" xmlns:local="clr-namespace:HuntingFarm.Windows" mc:Ignorable="d" Style="{StaticResource base\_window}" MinHeight="400" MinWidth="600" Width="600" WindowStartupLocation="CenterScreen"> <Grid > <Grid.RowDefinitions> <RowDefinition Height="40"/> <RowDefinition Height="\*"/> <RowDefinition Height="40"/> </Grid.RowDefinitions> <Grid Background="{StaticResource additional}"/> <TextBlock Text="Окно животных" HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="20" FontWeight="Bold"/> <Grid Background="{StaticResource additional}" Grid.Row="2"/> <ListBox x:Name="ListBoxAnimals" ScrollViewer.VerticalScrollBarVisibility="Visible" SelectedValuePath="id" Grid.Row="1" Grid.Column="0"> <ListBox.ItemTemplate> <DataTemplate> <Grid HorizontalAlignment="Stretch" UseLayoutRounding="True"> <Grid.ColumnDefinitions> <ColumnDefinition Width="100"/> <ColumnDefinition Width="\*"/> </Grid.ColumnDefinitions> <Image Width="100" Height="100" Grid.Column="0" Stretch="Uniform" Margin="5 2 5 2" Source="{Binding Path=GetAnimalImage}"/> <StackPanel Grid.Column="1" VerticalAlignment="Center"> <TextBlock Margin="2" FontSize="16" FontWeight="Bold" Text="{Binding Path=Name}" Width="auto" HorizontalAlignment="Left"/> <TextBlock Margin="2" FontSize="16" FontWeight="Bold" Text="{Binding Path=Description}" Width="auto" HorizontalAlignment="Left"/> <TextBlock Margin="2" FontSize="16" FontWeight="Bold" Text="{Binding Path=Difficulty.Name}" Width="auto" HorizontalAlignment="Left"/> </StackPanel> </Grid> </DataTemplate> </ListBox.ItemTemplate> </ListBox> <Button x:Name="btnAddOrEditAnimal" Content="Добавить/Изменить" Grid.Row="2" Click="btnAddOrEditAnimal\_Click"/> </Grid> </Window>  **Файл программного кода** using HuntingFarm.Models; 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 HuntingFarm.Windows { /// <summary> /// Логика взаимодействия для AnimalsWindow.xaml /// </summary> public partial class AnimalsWindow : Window { public AnimalsWindow() { InitializeComponent(); ListBoxAnimals.ItemsSource = HuntEntities.GetContext().Animals.OrderBy(P => P.Name).ToList(); } private void btnAddOrEditAnimal\_Click(object sender, RoutedEventArgs e) { if (ListBoxAnimals.SelectedItem != null) { AddAnimalWindow addAnimalWindow = new AddAnimalWindow(ListBoxAnimals.SelectedItem as Animal); addAnimalWindow.Owner = this; this.Hide(); addAnimalWindow.Show(); } else { AddAnimalWindow addAnimalWindow = new AddAnimalWindow(); addAnimalWindow.Show(); } } } }  **Окно Добавления/редактирования животного Файл интерфейса** <Window x:Class="HuntingFarm.Windows.AddAnimalWindow" xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation" xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" xmlns:d="http://schemas.microsoft.com/expression/blend/2008" xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006" xmlns:local="clr-namespace:HuntingFarm.Windows" mc:Ignorable="d" Title="Добавление животного" Height="350" Width="300" WindowStartupLocation="CenterScreen"> <Grid> <Grid.RowDefinitions> <RowDefinition Height="40"/> <RowDefinition Height="40"/> <RowDefinition Height="40"/> <RowDefinition Height="40"/> <RowDefinition Height="60"/> <RowDefinition Height="\*"/> </Grid.RowDefinitions> <Grid Background="{StaticResource additional}"/> <TextBlock Text="Добавление животного" HorizontalAlignment="Center" VerticalAlignment="Center" FontWeight="Bold"/> <StackPanel Orientation="Horizontal" Grid.Row="1" HorizontalAlignment="Left" VerticalAlignment="Center" Margin="10 0 0 0"> <TextBlock Text="Название:" Style="{StaticResource textblock\_textbox\_combo}" /> <TextBox x:Name="TbName" Width="150" /> </StackPanel> <StackPanel Grid.Row="2" Orientation="Horizontal" HorizontalAlignment="Left" VerticalAlignment="Center" Margin="10 0 0 0"> <TextBlock Text="Описание:" Style="{StaticResource textblock\_textbox\_combo}" /> <TextBox x:Name="TbDescription" Width="150" /> </StackPanel> <StackPanel Grid.Row="3" Orientation="Horizontal" HorizontalAlignment="Left" VerticalAlignment="Center" Margin="10 0 0 0"> <TextBlock Text="Сложность:" Style="{StaticResource textblock\_textbox\_combo}" /> <ComboBox DisplayMemberPath="Name" x:Name="ComboDifficulty" Width="150" /> </StackPanel> <StackPanel Grid.Row="4" Orientation="Horizontal"> <Button x:Name="btnLoadImage" Width="120" HorizontalAlignment="Left" Content="Загрузить фото" Click="btnLoadImage\_Click"/> <Image x:Name="ImagePreview" Height="60"/> </StackPanel> <StackPanel Orientation="Horizontal" Grid.Row="5" HorizontalAlignment="Center"> <Button Content="Добавить" Width="120" x:Name="btnOk" Click="btnOk\_Click"/> <Button Content="Отмена" Width="120" x:Name="btnCancle" Click="btnCancle\_Click"/> </StackPanel> </Grid> </Window>  **Файл программного кода** 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 HuntingFarm.Models; using System.Windows.Media; using System.Windows.Media.Imaging; using System.Windows.Shapes; using Microsoft.Win32; using System.IO; namespace HuntingFarm.Windows { /// <summary> /// Логика взаимодействия для AddAnimalPage.xaml /// </summary> public partial class AddAnimalWindow : Window { private string \_photoName = null, \_filePath = null, \_currentDirectory = Directory.GetCurrentDirectory() + @"\Images\Animals\", \_currentDirectoryNull = Directory.GetCurrentDirectory() + @"\Images\"; Animal \_currentAnimal; public AddAnimalWindow() { InitializeComponent(); ComboDifficulty.ItemsSource = HuntEntities.GetContext().Difficulties.OrderBy(p => p.Name).ToList(); } public AddAnimalWindow(Animal animal) { InitializeComponent(); var items = HuntEntities.GetContext().Difficulties.OrderBy(p => p.Name).ToList(); int index = -1; for (int i = 0; i < items.Count; i++) { if (items[i].id == animal.DifficultyId) index = i; } ComboDifficulty.ItemsSource = items; ComboDifficulty.SelectedIndex = index; \_currentAnimal = animal; string file = ""; if (\_currentAnimal.Image == null) file = \_currentDirectoryNull + "default.png"; else file = \_currentDirectory + \_currentAnimal.Image; ImagePreview.Source = new BitmapImage(new Uri(file)); TbName.Text = \_currentAnimal.Name; TbDescription.Text = \_currentAnimal.Description; ComboDifficulty.SelectedIndex = index; } private void btnOk\_Click(object sender, RoutedEventArgs e) { StringBuilder \_errors = CheckFields(); if (\_errors.Length > 0) { MessageBox.Show(\_errors.ToString()); return; } try { if (\_currentAnimal == null) { string photo = \_photoName == null ? null : ChangePhotoName(\_currentDirectory); string dest = \_currentDirectory + photo; if (\_photoName != null) File.Copy(\_filePath, dest); \_currentAnimal = new Animal() { Name = TbName.Text, Description = TbDescription.Text, DifficultyId = (ComboDifficulty.SelectedItem as Difficulty).id, Image = photo, }; HuntEntities.GetContext().Animals.Add(\_currentAnimal); } else { Animal animal = HuntEntities.GetContext().Animals.First(p => p.id == \_currentAnimal.id); animal.Name = TbName.Text; animal.Description = TbDescription.Text; animal.DifficultyId = (ComboDifficulty.SelectedItem as Difficulty).id; animal.Image = \_photoName; } HuntEntities.GetContext().SaveChanges(); MessageBox.Show("Успешно добавлено"); this.Close(); } catch (Exception ex) { MessageBox.Show(ex.Message); } } private void btnCancle\_Click(object sender, RoutedEventArgs e) { this.Close(); } private void btnLoadImage\_Click(object sender, RoutedEventArgs e) { try { OpenFileDialog op = new OpenFileDialog(); op.Title = "Выбор фото"; op.Filter = "JPEG Files (\*.jpeg)|\*.jpeg|PNG Files (\*.png)|\*.png|JPG Files (\*.jpg)|\*.jpg|GIF (\*.gif)|\*.gif"; if(op.ShowDialog() == true) { ImagePreview.Source = new BitmapImage( new Uri(op.FileName) ); \_photoName = op.SafeFileName; \_filePath = op.FileName; } } catch (Exception ex) { MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error); \_filePath = null; } } string ChangePhotoName(string \_currentDirectory) { string x = \_currentDirectory + \_photoName; string photoName = \_photoName; int i = 0; if (File.Exists(x)) { while (File.Exists(x)) { i++; x = \_currentDirectory + i.ToString() + photoName; } photoName = i.ToString() + photoName; } \_photoName = photoName; return photoName; } private StringBuilder CheckFields() { StringBuilder sb = new StringBuilder(); if (string.IsNullOrWhiteSpace(TbName.Text)) sb.AppendLine("Введите название животного"); if (string.IsNullOrWhiteSpace(TbDescription.Text)) sb.AppendLine("Введите описание"); if (ComboDifficulty.SelectedIndex == -1) sb.AppendLine("Выберите уровень сложности"); return sb; } } }  **Окно Добавления/редактирования дома Файл интерфейса** <Window x:Class="HuntingFarm.Windows.AddOrEditHouseWindow" xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation" xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" xmlns:d="http://schemas.microsoft.com/expression/blend/2008" xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006" xmlns:local="clr-namespace:HuntingFarm.Windows" mc:Ignorable="d" Title="Добавление дома" Height="300" Width="300" WindowStartupLocation="CenterScreen"> <Grid> <Grid.RowDefinitions> <RowDefinition Height="40"/> <RowDefinition Height="40"/> <RowDefinition Height="40"/> <RowDefinition Height="60"/> <RowDefinition Height="\*"/> </Grid.RowDefinitions> <Grid Background="{StaticResource additional}"/> <TextBlock Text="Добавление дома" HorizontalAlignment="Center" VerticalAlignment="Center" FontWeight="Bold"/> <StackPanel Orientation="Horizontal" Grid.Row="1" HorizontalAlignment="Left" VerticalAlignment="Center" Margin="10 0 0 0"> <TextBlock Text="Название:" Style="{StaticResource textblock\_textbox\_combo}" /> <TextBox x:Name="TbName" Width="150" /> </StackPanel> <StackPanel Grid.Row="2" Orientation="Horizontal" HorizontalAlignment="Left" VerticalAlignment="Center" Margin="10 0 0 0"> <TextBlock Text="Описание:" Style="{StaticResource textblock\_textbox\_combo}" /> <TextBox x:Name="TbDescription" Width="150"/> </StackPanel> <StackPanel Grid.Row="3" Orientation="Horizontal"> <Button x:Name="btnLoadImage" Width="120" HorizontalAlignment="Left" Content="Загрузить фото" Click="btnLoadImage\_Click"/> <Image x:Name="ImagePreview" Height="60"/> </StackPanel> <StackPanel Orientation="Horizontal" Grid.Row="5" HorizontalAlignment="Center"> <Button Content="Добавить" Width="120" x:Name="btnOk" Click="btnOk\_Click"/> <Button Content="Отмена" Width="120" x:Name="btnCancle" Click="btnCancle\_Click"/> </StackPanel> </Grid> </Window>  **Файл программного кода** using HuntingFarm.Models; using Microsoft.Win32; using System; using System.Collections.Generic; using System.IO; 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 HuntingFarm.Windows { /// <summary> /// Логика взаимодействия для AddOrEditHouseWindow.xaml /// </summary> public partial class AddOrEditHouseWindow : Window { private string \_photoName = null, \_filePath = null, \_currentDirectory = Directory.GetCurrentDirectory() + @"\Images\Houses\", \_currentDirectoryNull = Directory.GetCurrentDirectory() + @"\Images\"; House \_currentHouse; public AddOrEditHouseWindow() { InitializeComponent(); } public AddOrEditHouseWindow(House house) { InitializeComponent(); \_currentHouse = house; string file = ""; if (\_currentHouse.Image == null) file = \_currentDirectoryNull + "default.png"; else file = \_currentDirectory + \_currentHouse.Image; ImagePreview.Source = new BitmapImage(new Uri(file)); TbName.Text = \_currentHouse.Name; TbDescription.Text = \_currentHouse.Description; } private void btnLoadImage\_Click(object sender, RoutedEventArgs e) { try { OpenFileDialog op = new OpenFileDialog(); op.Title = "Выбор фото"; op.Filter = "JPEG Files (\*.jpeg)|\*.jpeg|PNG Files (\*.png)|\*.png|JPG Files (\*.jpg)|\*.jpg|GIF (\*.gif)|\*.gif"; if (op.ShowDialog() == true) { ImagePreview.Source = new BitmapImage(new Uri(op.FileName)); \_photoName = op.SafeFileName; \_filePath = op.FileName; } } catch (Exception ex) { MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error); \_filePath = null; } } string ChangePhotoName(string \_currentDirectory) { string x = \_currentDirectory + \_photoName; string photoName = \_photoName; int i = 0; if (File.Exists(x)) { while (File.Exists(x)) { i++; x = \_currentDirectory + i.ToString() + photoName; } photoName = i.ToString() + photoName; } \_photoName = photoName; return photoName; } private void btnOk\_Click(object sender, RoutedEventArgs e) { StringBuilder \_errors = CheckFields(); if (\_errors.Length > 0) { MessageBox.Show(\_errors.ToString()); return; } try { string photo = \_photoName == null ? null : ChangePhotoName(\_currentDirectory); if (\_currentHouse == null) { string dest = \_currentDirectory + photo; if (\_photoName != null) File.Copy(\_filePath, dest); \_currentHouse = new House() { Name = TbName.Text, Description = TbDescription.Text, Image = photo, }; HuntEntities.GetContext().Houses.Add(\_currentHouse); } else { House house = HuntEntities.GetContext().Houses.First(p => p.id == \_currentHouse.id); house.Name = TbName.Text; house.Description = TbDescription.Text; house.Image = \_photoName; } HuntEntities.GetContext().SaveChanges(); MessageBox.Show("Успешно добавлено"); this.Close(); } catch (Exception ex) { MessageBox.Show(ex.Message); } } private void btnCancle\_Click(object sender, RoutedEventArgs e) { this.Close(); } private StringBuilder CheckFields() { StringBuilder sb = new StringBuilder(); if (string.IsNullOrWhiteSpace(TbName.Text)) sb.AppendLine("Введите название животного"); if (string.IsNullOrWhiteSpace(TbDescription.Text)) sb.AppendLine("Введите описание"); return sb; } } }  **Окно Добавления/редактирования мероприятия Файл интерфейса** <Window x:Class="HuntingFarm.Windows.AddOrEditHuntEventWindow" xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation" xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" xmlns:d="http://schemas.microsoft.com/expression/blend/2008" xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006" xmlns:local="clr-namespace:HuntingFarm.Windows" mc:Ignorable="d" Title="Добавление мероприятие" Height="360" Width="320" WindowStartupLocation="CenterScreen"> <Grid> <Grid.RowDefinitions> <RowDefinition Height="40"/> <RowDefinition Height="40"/> <RowDefinition Height="40"/> <RowDefinition Height="40"/> <RowDefinition Height="40"/> <RowDefinition Height="60"/> <RowDefinition Height="\*"/> </Grid.RowDefinitions> <Grid Background="{StaticResource additional}"/> <TextBlock Text="Добавление мероприятие" HorizontalAlignment="Center" VerticalAlignment="Center" FontWeight="Bold"/> <StackPanel Orientation="Horizontal" Grid.Row="1" HorizontalAlignment="Left" VerticalAlignment="Center" Margin="10 0 0 0"> <TextBlock Text="Название:" Style="{StaticResource textblock\_textbox\_combo}" Width="150"/> <TextBox x:Name="TbName" Width="150" /> </StackPanel> <StackPanel Grid.Row="2" Orientation="Horizontal" HorizontalAlignment="Left" VerticalAlignment="Center" Margin="10 0 0 0"> <TextBlock Text="Животное:" Style="{StaticResource textblock\_textbox\_combo}" Width="150"/> <ComboBox x:Name="ComboAnimal" DisplayMemberPath="Name" Width="150" /> </StackPanel> <StackPanel Grid.Row="3" Orientation="Horizontal" HorizontalAlignment="Left" VerticalAlignment="Center" Margin="10 0 0 0"> <TextBlock Text="Дом:" Style="{StaticResource textblock\_textbox\_combo}" Width="150"/> <ComboBox DisplayMemberPath="Name" x:Name="ComboHouse" Width="150" /> </StackPanel> <StackPanel Grid.Row="4" Orientation="Horizontal" HorizontalAlignment="Left" VerticalAlignment="Center" Margin="10 0 0 0"> <TextBlock Text="Администратор:" Style="{StaticResource textblock\_textbox\_combo}" Width="150"/> <ComboBox DisplayMemberPath="Name" x:Name="ComboAdmin" Width="150" /> </StackPanel> <StackPanel Grid.Row="5" Orientation="Horizontal" HorizontalAlignment="Left" VerticalAlignment="Center" Margin="10 0 0 0"> <TextBlock Text="Стоимость:" Style="{StaticResource textblock\_textbox\_combo}" Width="150"/> <TextBox x:Name="TbCost" Width="150"/> </StackPanel> <StackPanel Orientation="Horizontal" Grid.Row="6" HorizontalAlignment="Center"> <Button Content="Добавить" Width="120" x:Name="btnOk" Click="btnOk\_Click"/> <Button Content="Отмена" Width="120" x:Name="btnCancle" Click="btnCancle\_Click"/> </StackPanel> </Grid> </Window> **Файл программного кода** 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; using HuntingFarm.Models; namespace HuntingFarm.Windows { /// <summary> /// Логика взаимодействия для AddOrEditHuntEventWindow.xaml /// </summary> public partial class AddOrEditHuntEventWindow : Window { Hunting \_currentHunting = null; public AddOrEditHuntEventWindow() { InitializeComponent(); ComboAnimal.ItemsSource = HuntEntities.GetContext().Animals.ToList(); ComboHouse.ItemsSource = HuntEntities.GetContext().Houses.ToList(); ComboAdmin.ItemsSource = HuntEntities.GetContext().Users.Where(p => p.RoleId == 2).ToList(); } public AddOrEditHuntEventWindow(Hunting hunting) { InitializeComponent(); \_currentHunting = hunting; List<Animal> animals = HuntEntities.GetContext().Animals.ToList(); List<House> houses = HuntEntities.GetContext().Houses.ToList(); List<User> users = HuntEntities.GetContext().Users.Where(p => p.RoleId == 2).ToList(); int index = -1; for (int i = 0; i < animals.Count; i++) if (animals[i].id == \_currentHunting.AnimalId) index = i; ComboAnimal.ItemsSource = animals; ComboAnimal.SelectedIndex = index; index = -1; for (int i = 0; i < houses.Count; i++) if (houses[i].id == \_currentHunting.HouseId) index = i; ComboHouse.ItemsSource = houses; ComboHouse.SelectedIndex = index; index = -1; for (int i = 0; i < users.Count; i++) if (users[i].id == \_currentHunting.AdminId) index = i; ComboAdmin.ItemsSource = users; ComboAdmin.SelectedIndex = index; TbName.Text = \_currentHunting.Name; TbCost.Text = \_currentHunting.Cost.ToString(); } private void btnOk\_Click(object sender, RoutedEventArgs e) { StringBuilder \_errors = CheckFields(); if (!decimal.TryParse(TbCost.Text, out decimal cost)) \_errors.AppendLine("Некоректная цена"); if (\_errors.Length > 0 ) { MessageBox.Show(\_errors.ToString()); return; } try { if (\_currentHunting == null) { \_currentHunting = new Hunting() { Name = TbName.Text, AnimalId = (ComboAnimal.SelectedItem as Animal).id, HouseId= (ComboHouse.SelectedItem as House).id, AdminId = (ComboAdmin.SelectedItem as User).id, Cost = cost, }; HuntEntities.GetContext().Huntings.Add(\_currentHunting); } else { Hunting hunting = HuntEntities.GetContext().Huntings.First(p => p.id == \_currentHunting.id); hunting.Name = TbName.Text; hunting.AnimalId = (ComboAnimal.SelectedItem as Animal).id; hunting.HouseId = (ComboHouse.SelectedItem as House).id; hunting.AdminId = (ComboAdmin.SelectedItem as User).id; hunting.Cost = decimal.Parse(TbCost.Text); } HuntEntities.GetContext().SaveChanges(); MessageBox.Show("Успешно добавлено"); this.Close(); } catch (Exception ex) { MessageBox.Show(ex.Message); } } private void btnCancle\_Click(object sender, RoutedEventArgs e) { this.Close(); } private StringBuilder CheckFields() { StringBuilder sb = new StringBuilder(); if (ComboAnimal.SelectedIndex == -1) sb.AppendLine("Выберите животного"); if (ComboHouse.SelectedIndex == -1) sb.AppendLine("Выберите дом"); if (ComboAdmin.SelectedIndex == -1) sb.AppendLine("Выберите администратора"); return sb; } } }  **Окно Профиля Файл интерфейса** <Window x:Class="HuntingFarm.Windows.ProfileWindow" xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation" xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" xmlns:d="http://schemas.microsoft.com/expression/blend/2008" xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006" xmlns:local="clr-namespace:HuntingFarm.Windows" mc:Ignorable="d" Title="Профиль" Style="{StaticResource base\_window}" Width="400" WindowStartupLocation="CenterScreen" Height="500" MinHeight="400" MinWidth="500"> <Grid> <Grid.RowDefinitions> <RowDefinition Height="40"/> <RowDefinition Height="20"/> <RowDefinition Height="20"/> <RowDefinition Height="\*"/> <RowDefinition Height="40"/> </Grid.RowDefinitions> <Grid.ColumnDefinitions> <ColumnDefinition Width="3\*"/> <ColumnDefinition Width="3\*"/> </Grid.ColumnDefinitions> <Grid Background="{StaticResource additional}" Grid.ColumnSpan="2"/> <TextBlock x:Name="TbTime" HorizontalAlignment="Center" Grid.Row="1" Grid.Column="1" VerticalAlignment="Center" Style="{StaticResource add\_window\_textblock}"/> <TextBlock x:Name="TbName" HorizontalAlignment="Center" Grid.Row="2" Grid.Column="1" VerticalAlignment="Center" Style="{StaticResource add\_window\_textblock}"/> <StackPanel Grid.Column="0" Grid.Row="3" VerticalAlignment="Center"> <Image x:Name="ImagePhoto" Height="200" Grid.Row="3"/> <Button x:Name="btnUploadPhoto" Content="Загрузить фото" Click="btnUploadPhoto\_Click"/> </StackPanel> <TextBlock Grid.Row="0" Grid.ColumnSpan="2" x:Name="TbWindow"/> <StackPanel Grid.Row="3" Grid.Column="1" VerticalAlignment="Center" x:Name="StackPanelAdmin" Visibility="Hidden"> <Button x:Name="BtnAddHuntEvents" Click="BtnAddHuntEvents\_Click" Content="Добавить мероприятие"/> <Button x:Name="BtnAddAnimal" Click="BtnAddAnimal\_Click" Content="Добавить животное"/> <Button x:Name="BtnAddHouse" Click="BtnAddHouse\_Click" Content="Добавить дом"/> <Button x:Name="BtnAddRole" Click="BtnAddRole\_Click" Content="Добавить роль"/> </StackPanel> <StackPanel Grid.Row="3" Grid.Column="1" VerticalAlignment="Center" x:Name="StackPanelUser" Visibility="Hidden"> <Button x:Name="BtnMyEvents" Click="BtnMyEvents\_Click" Content="Мои мероприятия"/> <Button x:Name="btnEditCurrentUser" Click="btnEditCurrentUser\_Click" Content="Изменить профиль"/> </StackPanel> </Grid> </Window>  **Файл программного кода** using HuntingFarm.Models; using Microsoft.Win32; using System; using System.Collections.Generic; using System.IO; using System.Linq; using System.Windows; using System.Windows.Media.Imaging; namespace HuntingFarm.Windows { /// <summary> /// Логика взаимодействия для ProfileWindow.xaml /// </summary> public partial class ProfileWindow : Window { private string \_photoName = null, \_filePath = null, \_currentDirectoryClient = Directory.GetCurrentDirectory() + @"\Images\Clients\", \_currentDirectoryAdministrator = Directory.GetCurrentDirectory() + @"\Images\Administrators\", \_currentDirectoryNull = Directory.GetCurrentDirectory() + @"\Images\"; public ProfileWindow() { InitializeComponent(); InitialWindow(); } void InitialWindow() { TbTime.Text = getTimeText(); TbName.Text = getUserName(); ImagePhoto.Source = new BitmapImage(new Uri(Manager.CurrentUser.GetPhoto, UriKind.Absolute)); if (Manager.CurrentUser.RoleId == 1) { StackPanelUser.Visibility = Visibility.Visible; } if (Manager.CurrentUser.RoleId == 2) { StackPanelAdmin.Visibility = Visibility.Visible; } } private string getTimeText() { DateTime timeNow = DateTime.Now; int hour = timeNow.Hour; int minute = timeNow.Minute; if (hour >= 9 && hour < 11 || hour == 11 && minute == 0) return "Доброе утро"; if (hour >= 11 && hour < 18 || hour == 18 && minute == 0) return "Добрый день"; if (hour >= 18 && hour < 24 || hour == 24 && minute == 0) return "Доброй вечер"; return "Доброй ночи"; } private string getUserName() { if (Manager.CurrentUser.GenderId == 1) return "Mr " + Manager.CurrentUser.Name + " " + Manager.CurrentUser.Patronymic + " " + Manager.CurrentUser.Surname; return "Ms " + Manager.CurrentUser.Name + " " + Manager.CurrentUser.Patronymic + " " + Manager.CurrentUser.Surname; } private void BtnAddHuntEvents\_Click(object sender, RoutedEventArgs e) { EventsWindow eventsWindow = new EventsWindow(); eventsWindow.Owner = this; this.Hide(); eventsWindow.Show(); } private void BtnAddAnimal\_Click(object sender, RoutedEventArgs e) { AnimalsWindow animalsWindow = new AnimalsWindow(); animalsWindow.Owner = this; this.Hide(); animalsWindow.Show(); } private void BtnAddHouse\_Click(object sender, RoutedEventArgs e) { HousesWindow housesWindow = new HousesWindow(); housesWindow.Owner = this; this.Hide(); housesWindow.Show(); } private void btnEditCurrentUser\_Click(object sender, RoutedEventArgs e) { RegistrationWindow registrationWindow = new RegistrationWindow(Manager.CurrentUser); registrationWindow.Owner = this; registrationWindow.Show(); } private void BtnMyEvents\_Click(object sender, RoutedEventArgs e) { AccountingsClientWindow accountingsClientWindow = new AccountingsClientWindow(); accountingsClientWindow.Owner = this; accountingsClientWindow.Show(); } private void BtnAddRole\_Click(object sender, RoutedEventArgs e) { UsersWindow usersWindow = new UsersWindow(); usersWindow.Show(); usersWindow.Owner = this; this.Hide(); } private void btnUploadPhoto\_Click(object sender, RoutedEventArgs e) { try { OpenFileDialog op = new OpenFileDialog(); op.Title = "Выбор фото"; op.Filter = "JPEG Files (\*.jpeg)|\*.jpeg|PNG Files (\*.png)|\*.png|JPG Files (\*.jpg)|\*.jpg|GIF (\*.gif)|\*.gif"; if (op.ShowDialog() == true) { ImagePhoto.Source = new BitmapImage(new Uri(op.FileName)); \_photoName = op.SafeFileName; \_filePath = op.FileName; } string diretory = Manager.CurrentUser.RoleId == 2 ? \_currentDirectoryAdministrator : \_currentDirectoryClient; string photo = \_photoName == null ? null : ChangePhotoName(diretory); string dest = diretory + photo; if (\_photoName != null) File.Copy(\_filePath, dest); User user = HuntEntities.GetContext().Users.First(p => p.id == Manager.CurrentUser.id); user.Image = \_photoName; HuntEntities.GetContext().SaveChanges(); MessageBox.Show("Успешно добавлено"); } catch (Exception ex) { MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error); \_filePath = null; } } private void btnUpdate\_Click(object sender, RoutedEventArgs e) { ImagePhoto.Source = new BitmapImage(new Uri(Manager.CurrentUser.GetPhoto, UriKind.Absolute)); } string ChangePhotoName(string \_currentDirectory) { string x = \_currentDirectory + \_photoName; string photoName = \_photoName; int i = 0; if (File.Exists(x)) { while (File.Exists(x)) { i++; x = \_currentDirectory + i.ToString() + photoName; } photoName = i.ToString() + photoName; } \_photoName = photoName; return photoName; } } }  **Окно Учет мероприятий Файл интерфейса** <Window x:Class="HuntingFarm.Windows.AccountingsClientWindow" xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation" xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" xmlns:d="http://schemas.microsoft.com/expression/blend/2008" xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006" xmlns:local="clr-namespace:HuntingFarm.Windows" mc:Ignorable="d" Title="AccountingsClientPage" MinHeight="400" Height="400" MinWidth="700" Width="700" WindowStartupLocation="CenterScreen" Style="{StaticResource base\_window}"> <Grid> <Grid.RowDefinitions> <RowDefinition Height="40"/> <RowDefinition Height="\*"/> <RowDefinition Height="40"/> </Grid.RowDefinitions> <TextBlock x:Name="TbHeader" HorizontalAlignment="Center" VerticalAlignment="Center"/> <Grid Background="{StaticResource additional}"/> <Grid Background="{StaticResource additional}" Grid.Row="2"/> <ListBox Grid.Row="1" x:Name="ListBoxEvents"> <ListBox.ItemTemplate> <DataTemplate> <Grid HorizontalAlignment="Stretch" UseLayoutRounding="True"> <Grid.ColumnDefinitions> <ColumnDefinition Width="100"/> <ColumnDefinition Width="\*"/> <ColumnDefinition Width="150"/> </Grid.ColumnDefinitions> <Image Width="100" Height="100" Grid.Column="0" Stretch="Uniform" Margin="5 2 5 2" Source="{Binding Path=Hunting.Animal.GetAnimalImage}"/> <StackPanel Grid.Column="1" VerticalAlignment="Center" Width="400"> <TextBlock Margin="2" FontSize="16" FontWeight="Bold" Text="{Binding Path=Hunting.Name}" Width="auto" HorizontalAlignment="Left"/> <TextBlock Margin="2" FontSize="16" FontWeight="Bold" Text="{Binding Path=Hunting.Animal.Name}" Width="auto" HorizontalAlignment="Left"/> <TextBlock Margin="2" FontSize="16" FontWeight="Bold" Text="{Binding Path=Date}" Width="auto" HorizontalAlignment="Left"/> </StackPanel> </Grid> </DataTemplate> </ListBox.ItemTemplate> </ListBox> <Button Grid.Row="2" Content="Удалить" x:Name="btnDeleteEvent" Click="btnDeleteEvent\_Click"/> </Grid> </Window>  **Файл программного кода** using HuntingFarm.Models; 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 HuntingFarm.Windows { /// <summary> /// Логика взаимодействия для AccountingsClientPage.xaml /// </summary> public partial class AccountingsClientWindow : Window { public AccountingsClientWindow() { InitializeComponent(); ListBoxEvents.ItemsSource = HuntEntities.GetContext().AccountingEvents.Where(p => p.ClientId == Manager.CurrentUser.id).ToList(); } private void btnDeleteEvent\_Click(object sender, RoutedEventArgs e) { if (ListBoxEvents.SelectedItem != null) { MessageBoxResult x = MessageBox.Show("Вы уверены, что хотите отписаться от мероприятия?", "Отписаться", MessageBoxButton.OKCancel, MessageBoxImage.Question); if(x == MessageBoxResult.OK) { HuntEntities.GetContext().AccountingEvents.Remove(ListBoxEvents.SelectedItem as AccountingEvent); MessageBox.Show("Успешно удалено"); } HuntEntities.GetContext().SaveChanges(); Close(); } else { MessageBox.Show("Мероприятие не выбрано"); } } } }  **Окно Выбора даты Файл интерфейса** <Window x:Class="HuntingFarm.Windows.SelectDateWindow" xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation" xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" xmlns:d="http://schemas.microsoft.com/expression/blend/2008" xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006" xmlns:local="clr-namespace:HuntingFarm.Windows" mc:Ignorable="d" Title="Выбор даты" Height="250" Width="300" MinHeight="250" MinWidth="300" WindowStartupLocation="CenterScreen" Style="{StaticResource base\_window}"> <Grid> <Grid.RowDefinitions> <RowDefinition Height="30"/> <RowDefinition Height="80"/> <RowDefinition Height="\*"/> </Grid.RowDefinitions> <Grid Background="{StaticResource additional}"/> <TextBlock Text="Ввод даты" HorizontalAlignment="Center" VerticalAlignment="Center" FontWeight="Bold"/> <DatePicker x:Name="datePicker" Grid.Row="1" VerticalAlignment="Center"/> <Button x:Name="btnAddAccount" Grid.Row="2" Content="Ок" Width="100" Click="btnAddAccount\_Click"/> </Grid> </Window>  **Файл программного кода** 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; using HuntingFarm.Models; using HuntingFarm.Pages; namespace HuntingFarm.Windows { /// <summary> /// Логика взаимодействия для SelectDateWindow.xaml /// </summary> public partial class SelectDateWindow : Window { private int \_huntId; public SelectDateWindow(int HuntId) { InitializeComponent(); \_huntId = HuntId; } private void btnAddAccount\_Click(object sender, RoutedEventArgs e) { if (datePicker.SelectedDate != null) { AccountingEvent hunting = new AccountingEvent() { HuntId = \_huntId, ClientId = Manager.CurrentUser.id, Date = (DateTime)datePicker.SelectedDate, }; HuntEntities.GetContext().AccountingEvents.Add(hunting); HuntEntities.GetContext().SaveChanges(); MessageBox.Show("Успешно"); Close(); } else { MessageBox.Show("Необходимо выбрать дату"); } } } }

**Приложение B** **Диск с исполняемым кодом программного модуля**