МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ

РОССИЙСКАЯ ФЕДЕРАЦИЯ

Федеральное государственное автономное образовательное учреждение

высшего образования

«ТЮМЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ НАУК

КАФЕДРА ИНФОРМАЦИОННЫХ СИСТЕМ

КУРСОВАЯ РАБОТА

по дисциплине «Проектирование информационных систем»

«Реестр объявлений о пропаже домашнего животного»

Исполнитель:

студентка Исмакова Д.К.

группа 189-1

Руководитель:

ассистент кафедры

программной и системной

инженерии Спрысков А.А.

СОДЕРЖАНИЕ

[1. ВВЕДЕНИЕ 3](#_Toc62348594)

[1.1. Общие сведения о предметной области 3](#_Toc62348595)

[1.2. Назначения, цели и задачи создания системы 4](#_Toc62348596)

[2. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 5](#_Toc62348597)

[2.1. Описание прецедентов 5](#_Toc62348598)

[2.2. Нефункциональные требования 6](#_Toc62348599)

[2.3. Модель предметной области 8](#_Toc62348600)

[3. ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ 10](#_Toc62348601)

[3.1. Системные диаграммы последовательностей 10](#_Toc62348602)

[3.2. Диаграммы коммуникаций 12](#_Toc62348603)

[3.3. Диаграмма классов проектирования 13](#_Toc62348604)

[4. РЕАЛИЗАЦИЯ 15](#_Toc62348605)

[4.1. Схема данных 15](#_Toc62348606)

[4.2. Пользовательский интерфейс 16](#_Toc62348607)

[4.3. Инструментарий разработки 19](#_Toc62348608)

[ЗАКЛЮЧЕНИЕ 20](#_Toc62348609)

[СПИСОК ЛИТЕРАТУРЫ И ССЫЛОК 21](#_Toc62348610)

# ВВЕДЕНИЕ

## Общие сведения о предметной области

В процессах учёта, участвуют различные организации, которые в рамках своей работы ведут свои соответствующие журналы учёта и хранят документы, подтверждающие мероприятие, проведённое с животным.

Таким образом, у каждого учреждения сформирован и постоянно пополняется в электронном и/или бумажном виде массив учётных записей и архив документов в объёме мероприятий, проведённых соответствующим учреждением, то есть каждое учреждение владеет той частью информации о животном, которая входит в сферу его компетенций по работе с животными.

Такая ситуация не позволяет найти и посмотреть:

* всю историю мероприятий, включая документы, определить владельца потерявшегося животного с идентификационной меткой;
* определить перечень проведённых мероприятий и их результаты как в целом, так и по каждому муниципальному образованию, и по каждому виду учреждений.

Все это создает трудности для граждан в получении актуальной и единой информации, в том числе:

* получить список организаций в сфере поставки товаров и оказания услуг для животных;
* подать и посмотреть объявления о пропаже/находке домашнего животного;
* найти питомца из перечня животных, содержащихся в приютах;
* получить владельцу домашнего животного доступ к электронному паспорту.

Именно поэтому необходимо создать целостную систему для устранения вышеперечисленных проблем.

## Назначения, цели и задачи создания системы

Создаваемая система предназначена для осуществления ведения целостного цифрового реестра объявлений о пропаже домашних животных, в частности:

* просмотра данных о животных,
* редактирование и внесение информации о новом животном с закреплением за определённой организацией,
* выполнение поисковых запросов пользователей с использованием различных критериев.

С точки зрения описания предмета автоматизации «как должно быть» основой процессов учёта домашних животных и животных без владельцев и процесса информирования населения должна стать база данных, единая для всех участников процессов.

Все сведения о животном с владельцем и без владельцев должны регистрироваться сотрудниками объектов автоматизации в БД животных, включая электронные образы документов, сопровождающие процессы учёта.

Таким образом, целями создания системы реестра являются:

* создание системы реестра единого для различных организаций и округов,
* сокращение времени работы с системой;
* увеличение количества пользователей, пользующихся системой.

Для выполнения этих целей необходимо выполнить следующие задачи:

* вывод данных о животных в реестр,
* возможность удобно просматривать записи, сюда входит создание фильтрации и сортировки,
* добавление новых данных о животных в реестр,
* удаление данных о животных,
* хранение реестра в базе данных.

# АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

## Описание прецедентов

Далее будут представлены прецеденты по теме «Реестр объявлений о пропаже домашнего животного», описывающие задачи системы совместно со сценарием использования.

**Прецедент 1.** Просмотр/поиск записей реестра

**Основной исполнитель.** Любой, кому доступен реестр

**Предусловия.** Пользователь аутентифицирован под любой ролью, кроме роли «Владелец»

**Основной процесс.**

|  |  |
| --- | --- |
| Действия исполнителя | Отклик системы |
| 1. Открывает реестр | 1. Определяет принадлежность к конкретной организации или округу |
|  | 1. Выводит доступные записи |
| 1. extend “Вывод записей в представлении” | |
| 1. extend “Сортировка записей” | |
| 1. extend “Фильтрация записей” | |
| 1. Просматривает записи |  |

**Постусловия.**

Записи просмотрены/найдены

**Прецедент 2.** Сортировка записей

**Основной исполнитель.** Любой, кому доступен реестр

**Предусловия.** Выведены записи реестра

**Основной процесс.**

|  |  |
| --- | --- |
| Действия исполнителя | Отклик системы |
| 1. Задает условия сортировки | 1. Применяет выбранные условия |
|  | 1. Выводит доступные записи |

**Постусловия.** Выведены записи с сортировкой

**Прецедент 3.** Фильтрация записей

**Основной исполнитель.** Любой, кому доступен реестр

**Предусловия.** Выведены записи реестра

**Основной процесс.**

|  |  |
| --- | --- |
| Действия исполнителя | Отклик системы |
| 1. Задает условия фильтрации | 1. Применяет выбранные условия |
|  | 1. Выводит доступные записи |

**Постусловия.** Выведены записи с фильтрацией

**Прецедент 4.** Вывод записей в представлении

**Основной исполнитель.** Любой, кому доступен реестр

**Предусловия.** Выведены записи реестра

**Основной процесс.**

|  |  |
| --- | --- |
| Действия исполнителя | Отклик системы |
| 1. Задает представление записей | 1. Применяет выбранное представление |
|  | 1. Выводит доступные записи в представлении |

**Постусловия.** Выведены записи в определенном представлении

## Нефункциональные требования

***Требования к режимам функционирования системы.***

Система должна функционировать в двух режимах:

* публичном, доступном всем неавторизованным пользователям и авторизованным пользователям под ролью «Владелец»;
* приватном, доступном авторизованным пользователям в соответствии с их ролью.

***Допустимые пределы модернизации и развития системы.***

Система должна обеспечить возможность модернизации и развития при изменении параметров объекта автоматизации, а также при

необходимости изменения состава требований к выполняемым функциям и видам обеспечения.

При модернизации информационного обеспечения должно быть обеспечено сохранение базовой структуры данных, которая обеспечивает переход на новую версию системы без потерь данных.

***Требования к эргономике и технической эстетике.***

Система должна иметь доступный и понятный графический интерфейс с простым дизайном, обеспечивающим ясное понимание работы программы.

***Требования к персоналу.***

Система должна обеспечивать возможность предоставления доступа только сотрудникам, которые назначены на роль и в соответствии с уровнем доступа его роли.

Пользователи должны иметь опыт работы с системой Windows 10 и иметь навыки работы с браузерными клиентскими приложениями.

В табл. 1 представлены роли пользователей, у которых имеется доступ к реестру.

Таблица 1. Роли пользователей

|  |  |  |
| --- | --- | --- |
| **Роль** | **Доступен для:** | **Реестр доступен:** |
| Администратор данных | удаления некорректных записей | в рамках всех записей реестра |
| Куратор ВетСлужбы | поиска и просмотра | в рамках всех записей реестра |
| Куратор ОМСУ | поиска и просмотра | в рамках записей, относящихся к соответствующему муниципальному образованию |
| Куратор по отлову | поиска и просмотра | в рамках записей, относящихся к соответствующей организации по отлову |
| Оператор ВетСлужбы | поиска и просмотра | в рамках всех записей реестра |
| Оператор ОМСУ | поиска и просмотра | в рамках записей, относящихся к соответствующему муниципальному образованию |
| Подписант ВетСлужбы | поиска и просмотра | в рамках всех записей реестра |
| Подписант ОМСУ | поиска и просмотра | в рамках записей, относящихся к соответствующему муниципальному образованию |
| Подписант по отлову | поиска и просмотра | в рамках записей, относящихся к соответствующей организации по отлову |

***Требования к защите информации.***

Требуется, чтобы пользователь имел ограниченный доступ к информации, согласно уровню его доступа.

***Требования к дополнительному ПО.***

Для корректной работы системы требуется предварительно установить следующее программное обеспечение:

* Windows Server 2019;
* NET Framework 4.6.1 и выше;
* MS SQL 2019;
* Entity Framework 6.

## Модель предметной области

*Модель предметной области —* это *визуальное* представление концептуальных классов или объектов реального мира в терминах предметной области.

Используя метод выделения существительных, были определены следующие слова, которые далее будут представлять концептуальные классы: пользователь, роль, организация, округ, реестр, запись, где под записью подразумевается карточка животного. Получаем следующие классы: «Пользователь», «Роль», «Организация», «Округ», «Реестр», «Карточка животного», «Животное». Сразу можно выделить отношения «Реестр» и «Пользователь», где пользователь взаимодействует с реестром.

В зависимости от роли, организации или округа, которые также находятся в отношении с пользователем, реестр предоставляет конкретный набор карточек животных для конкретного пользователя. Пользователь под определенной ролью может добавлять, удалять карточки животного. В одном реестре может содержаться множество карточек, и определенная карточка может содержаться только в одном реестре. У одного пользователя может быть только одна роль, организация и округ. Карточка животного содержит информацию только об одном животном.

На рис. 1 представлена модель предметной области, где концептуальные классы имеют атрибуты и отношения между собой.



Рисунок . Модель предметной области

# ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ

## Системные диаграммы последовательностей

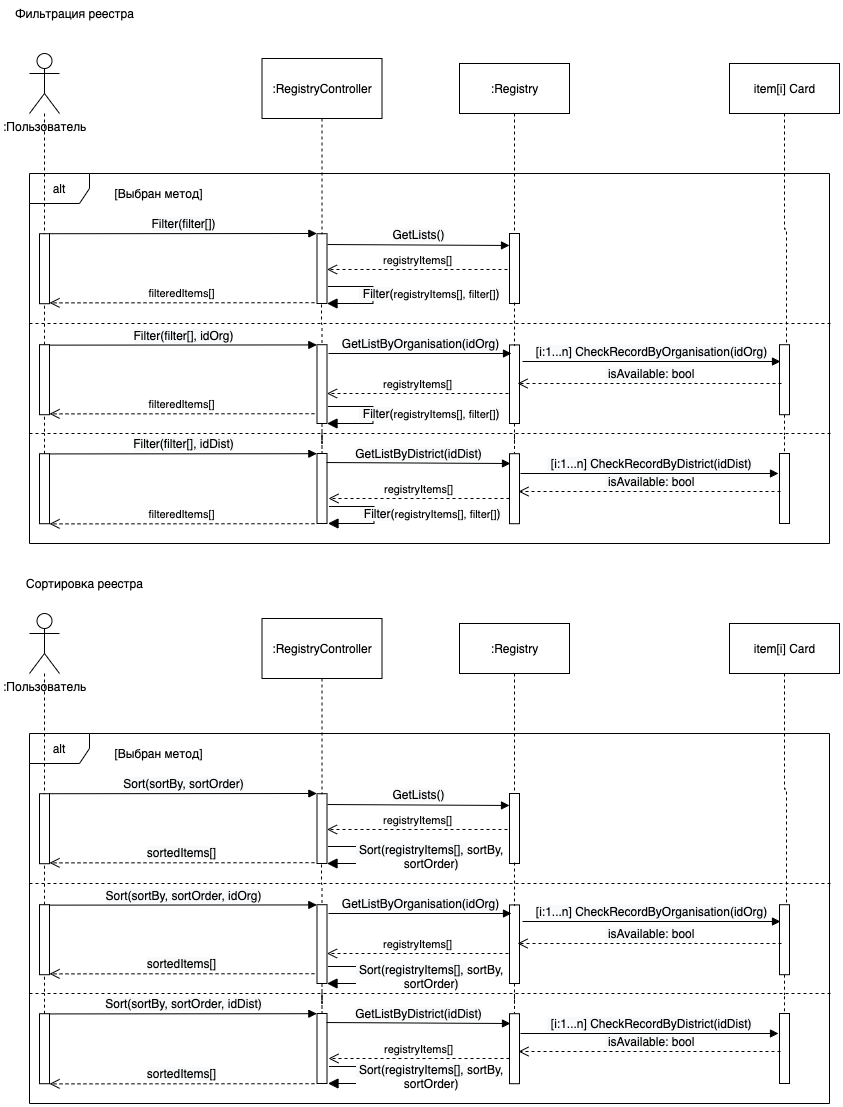
*Системная диаграмма последовательностей* — это схема, которая для определённого сценария прецедента показывает генерируемые внешними исполнителями события, их порядок. При этом вся система рассматривается как «чёрный ящик». Назначение данной диаграммы — отображение событий, передаваемых исполнителями системе через её границы.

Пример на рис. 2 отображает прецедент «Просмотр/поиск записей реестра». Пользователь, не имеющий организации и округа может просматривать все записи. Так как у всех пользователей, кроме администратора, есть принадлежность к какой-либо организации или округу, то соответственно только администратор может просматривать все записи. Остальные пользователи видят карточки животных, которые доступны по их роли.



Рисунок . Системная диаграмма последовательностей "Просмотр/поиск записей реестра"

Рисунок . Системные диаграммы последовательностей: "Фильтрация реестра", "Сортировка реестра"



На рис. 3 показаны системные диаграммы последовательностей соответствующие прецедентам: «Фильтрация реестра», «Сортировка реестра».

Фильтрация предполагает получение массива с фильтрами (filter[]), где пользователь инициирует событие Filter и в зависимости от переданных параметров получает отфильтрованные записи доступные конкретной роли. Для сортировки записей пользователю необходимо задать сортировку, создав событие Sort(sortBy, sortOrder) с дополнительным параметром в виде идентификатора организации или округа.

## Диаграммы коммуникаций

Диаграммы коммуникаций иллюстрируют взаимодействие объектов в формате графа или сети. Диаграммы коммуникаций, изображённые на рис. 4 можно рассматривать следующим образом. Диаграмма «Просмотр/поиск записей реестра» начинается с действия пользователя Open далее в зависимости от принадлежности пользователя к организации или округу отправляется сообщение GetLists, GetListByOrganisation или GetListByDistrict. Для сообщений GetListByOrganisation и GetListByDistrict проверяется доступность карточки с помощью сообщений CheckRecordByOrganisation и CheckRecordByDistrict соответственно. Возвращаемое булево значение позволяет вывести только доступные пользователю записи. Далее пользователь получает все доступные для него записи registryItems.

Фильтрация и сортировка реестра работают подобным образом. Пользователь посылает сообщение Filter с условиями сортировки, далее получаем записи как в диаграмме «Просмотр/поиск записей реестра» и уже в самом RegistryController производится фильтрация, после чего пользователю отправляется отфильтрованные записи filteredItems.

При сортировке пользователь отправляет сообщение Sort с условиями сортировки, далее получает записи доступные пользователю (см. диаграмму «Просмотр/поиск записей реестра») и в RegistryController производится сортировка. Пользователь получает отсоритрованные записи sortedItems.

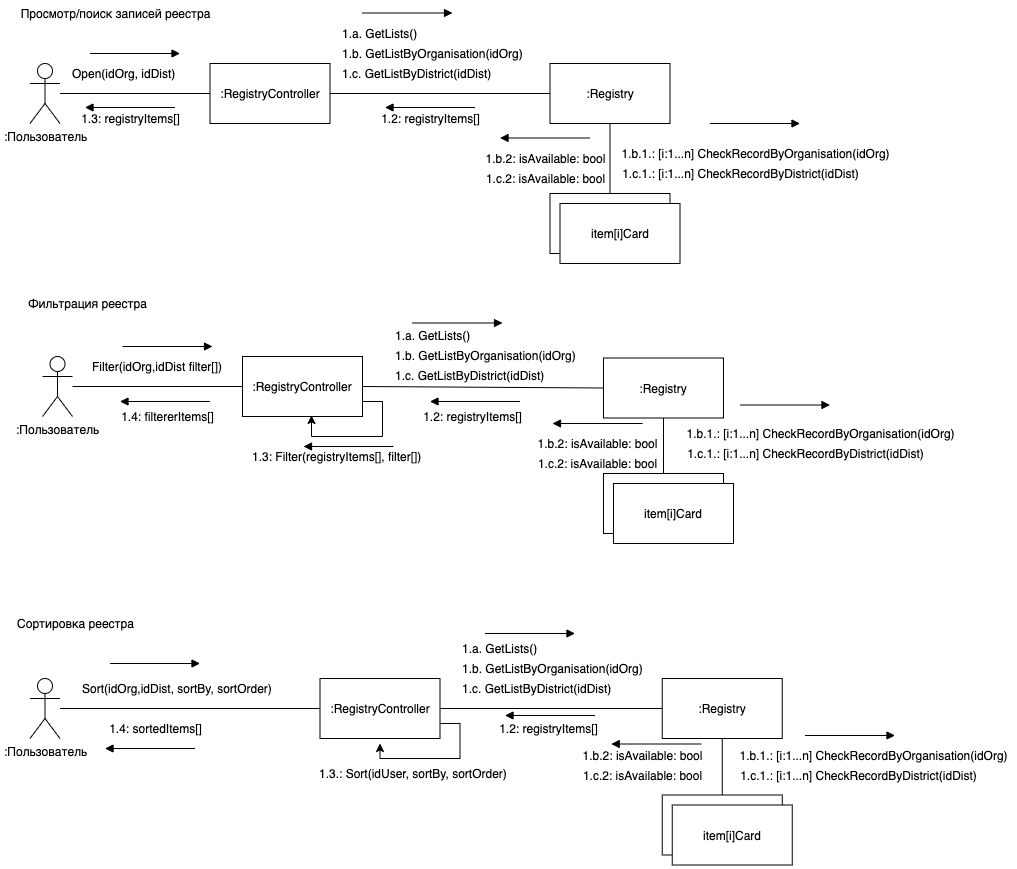


Рисунок . Диаграммы коммуникаций

## Диаграмма классов проектирования

Диаграмма классов проектирования, демонстрирует общую структуру иерархии классов системы, их коопераций, атрибутов (полей), методов, интерфейсов и взаимосвязей между ними.

Диаграмма классов на рис. 5 отображает взаимодействие классов и структуру в создаваемом ПО, а именно:

Класс User находится в отношении «агрегация» с классами Role, Organisation, District, что позволяет Пользователю иметь роль, организацию и округ.

Класс Registry имеет связь с Card, где Registry хранит список карточек для отображения через RegistryController. Также класс Card имеет отношение с классом Animal, для каждого животного заводится новая карточка.

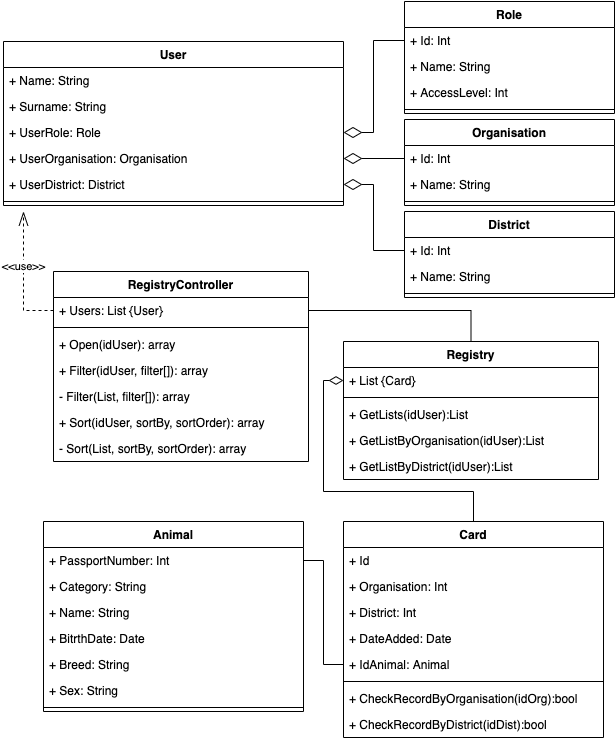


Рисунок . Диаграмма классов проектирования

# РЕАЛИЗАЦИЯ

## Схема данных

Данные о публикациях, животных, пользователях, их ролях, организациях и округах, а также о карточках, содержащих данные о животных должны где-то храниться. Именно для этого была спроектирована база данных, обеспечивающая целостность хранимой информации и ее безопасность.

В качестве базы данных была выбрана реляционная СУБД MySQL, а схема данных представлена на рис. 6.

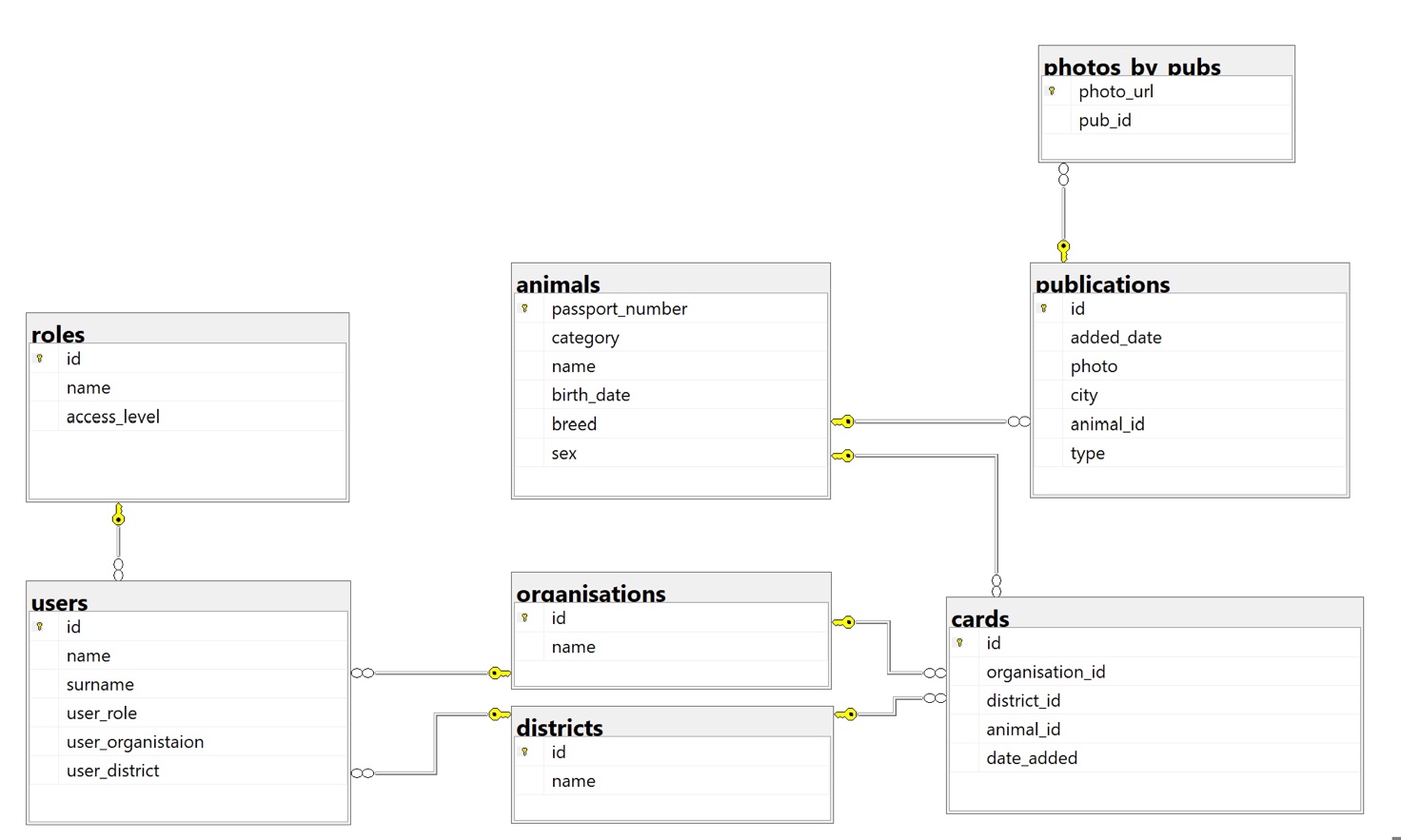


Рисунок . Схема данных

База данных для ПО состоит из 8-ми таблиц.

Таблица users хранит в себе данные о пользователях: имя, фамилия, также связана с таблицами roles (для получения роли пользователя с уровнем доступа access\_level), organisations (с данными об организации, в нашем случае название организации name) и districts (с данными об округе, которому принадлежит пользователь).

Карты животных хранятся в таблице cards, где указывается ID организации, ID округа, ID животного и дата добавления животного в данную карту. В свою очередь все данные о животных хранятся с таблице animals, а именно номер паспорта, категория животного, имя, дата рождения, порода и пол.

Таблица объявлений publications хранит в себе данные о публикациях животных с прикрепленными к ним фотографиями из таблицы photos\_by\_pubs.

## Пользовательский интерфейс

Страница реестра представляет собой таблицу с возможностью фильтрации и сортировки, на рис.7 представлен вид реестра для пользователя, относящегося к организации. Функционал такого реестра ограничен, так как пользователь может только просматривать, фильтровать, сортировать и добавлять новые карточки животных, тогда как администратор может еще и удалять (рис. 8). Для уточнения, данный реестр является закрытым, то есть доступен пользователям только с ролями, принадлежащими организациям, округам или являющимися администраторами.

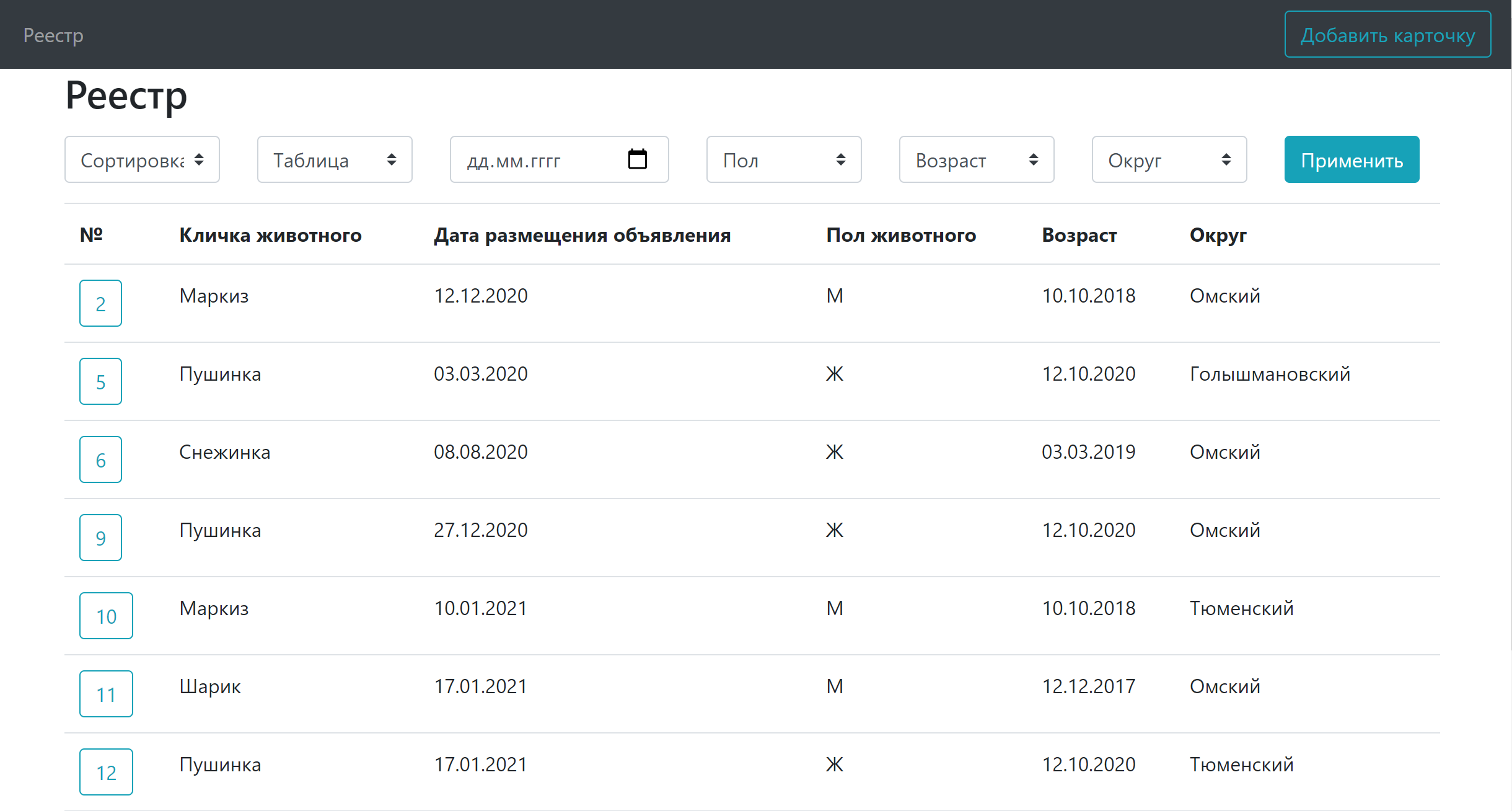


Рисунок . Реестр, ограниченный функционал

Для того, чтобы отфильтровать, отсортировать или вывести в определенном представлении необходимо выбрать параметры и нажать «Применить». На рис. 9 показан вывод записей в представлении «Карточки» под учетной записью администратора.

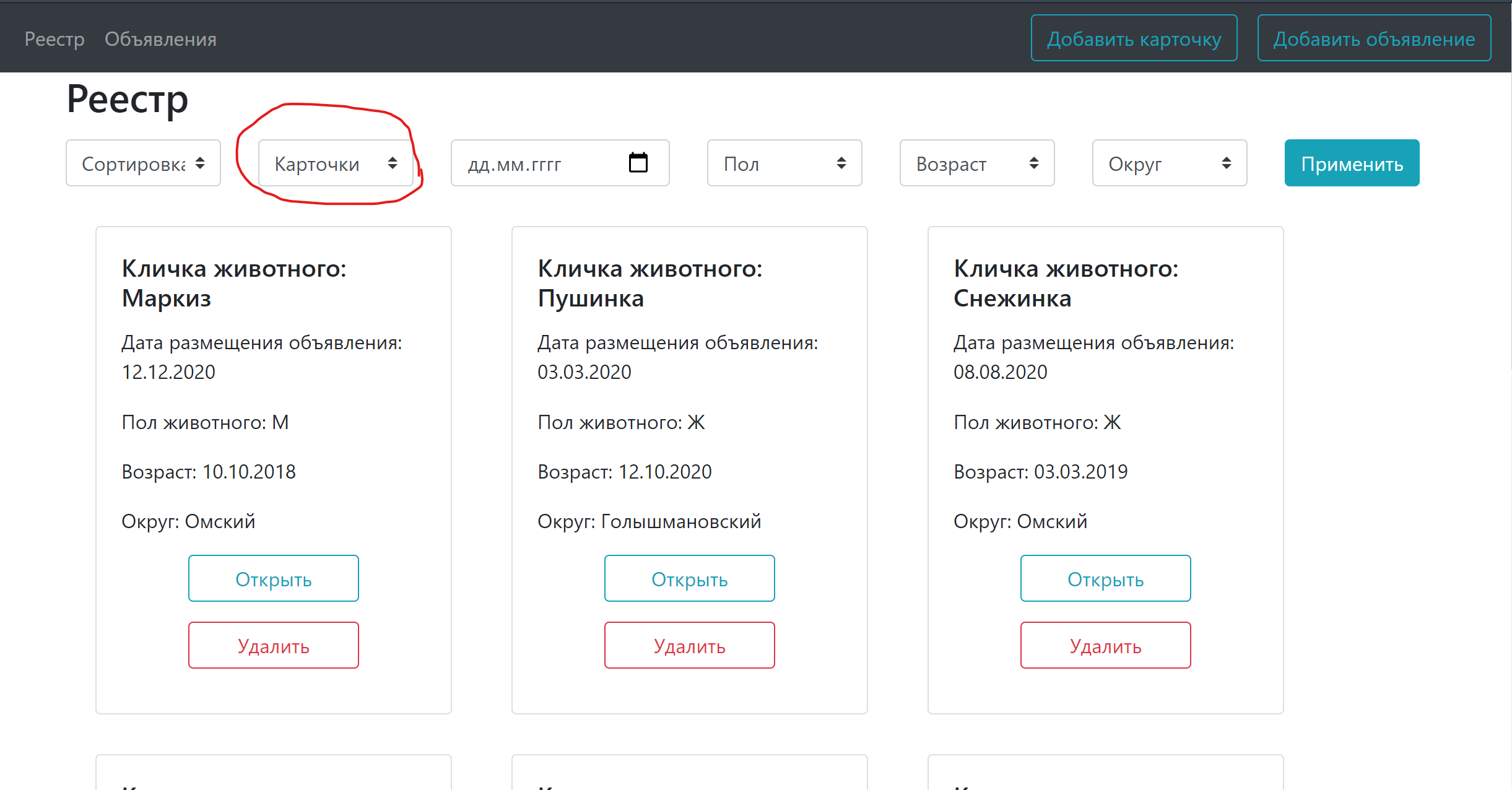


Рисунок 9. Вывод записей реестра в представлении "Карточки"

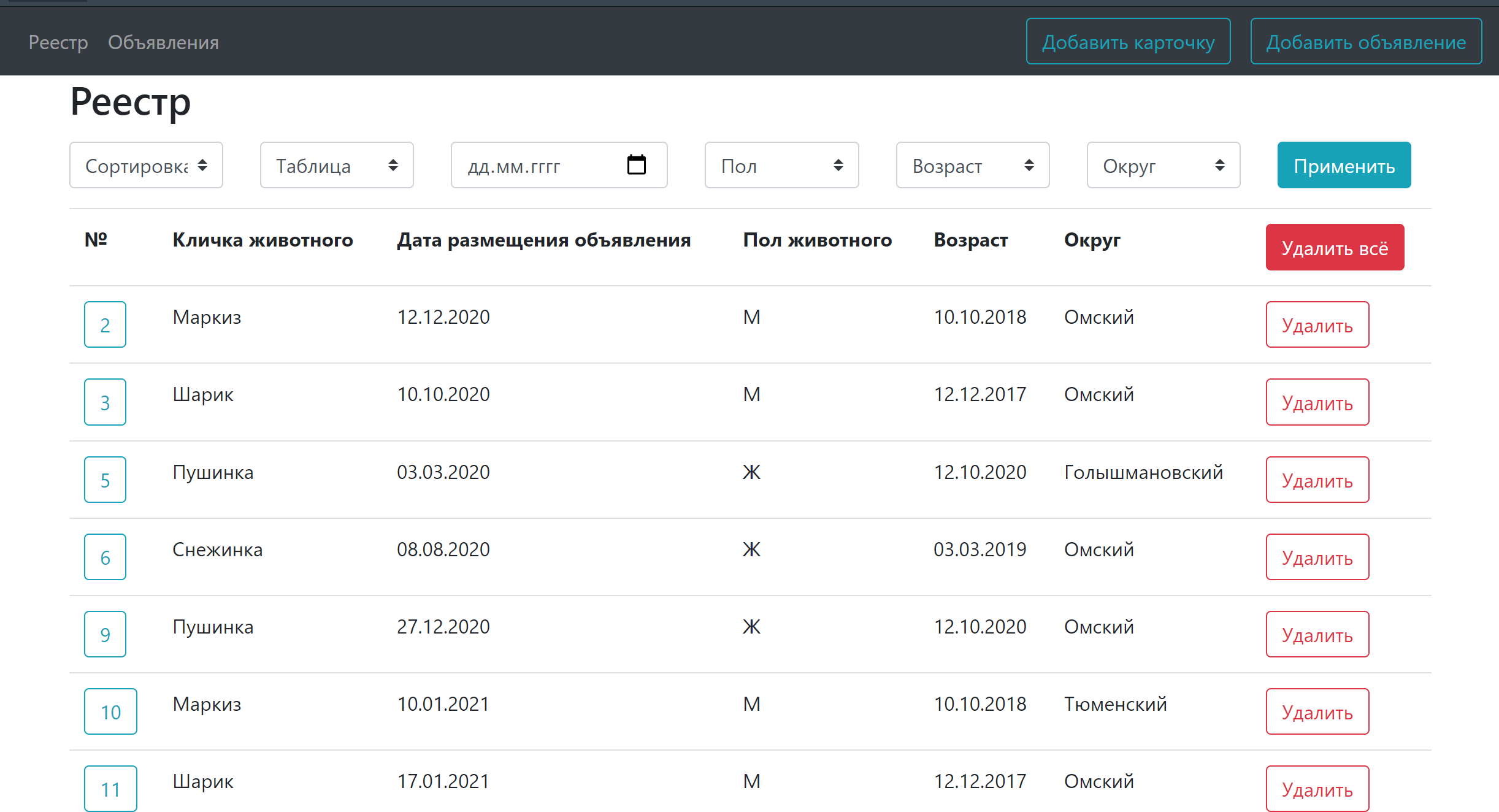


Рисунок 8. Реестр, полный функционал

Для добавлении новой карточки в верхнем правом углу экрана необходимо нажать «Добавить карточку», после этого появится страница для добавления карточки животного (рис. 10, 11). После нажатия на «Добавить» карта животного появится в реестре.

Удалять записи закрытого реестра может только администратор, для этого необходимо нажать «Удалить» рядом с конкретной записью и запись будет удалена (рис. 12).



Рисунок 12. Удаление записей

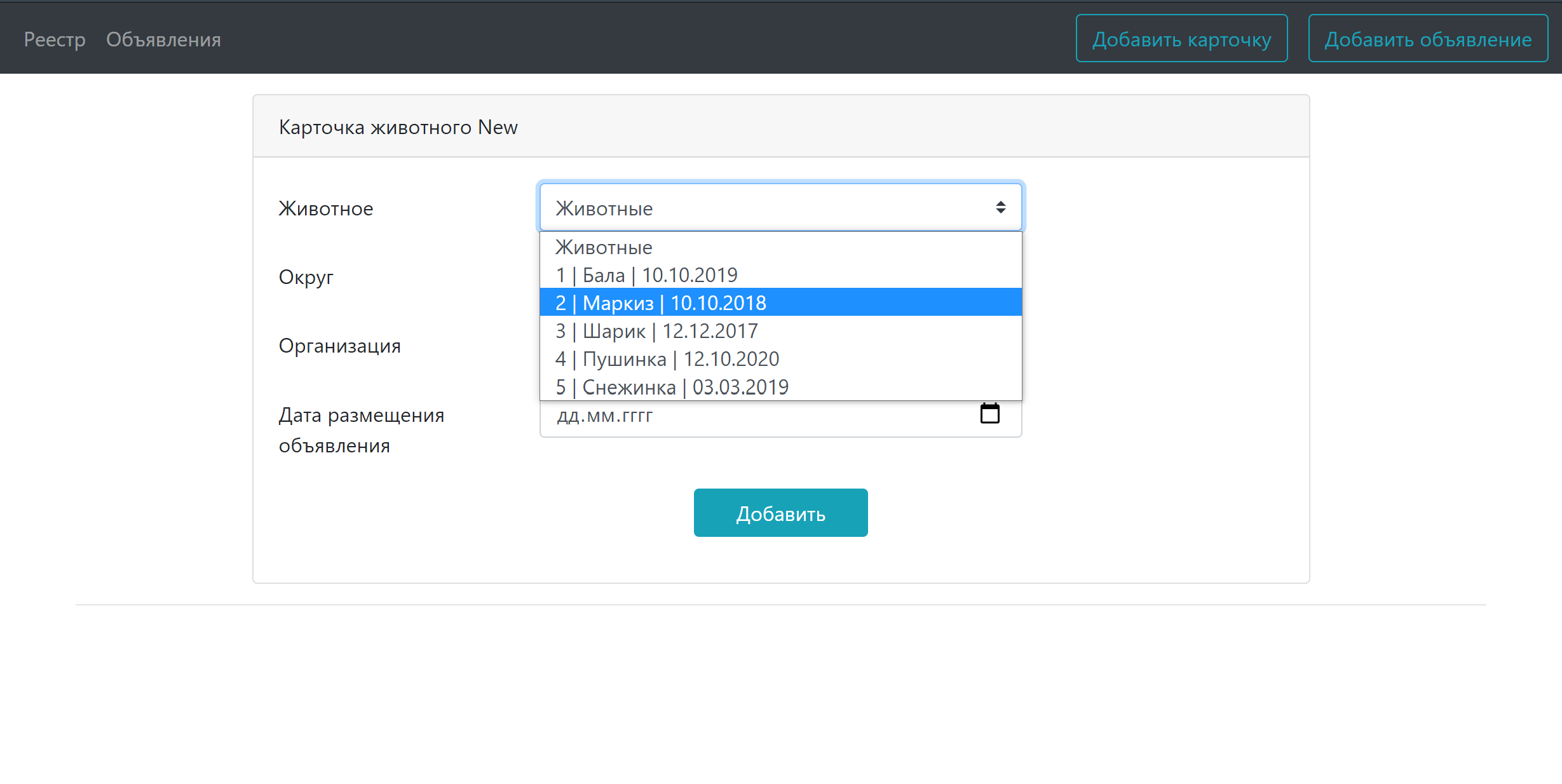


Рисунок 11. Страница для добавления карточки животного

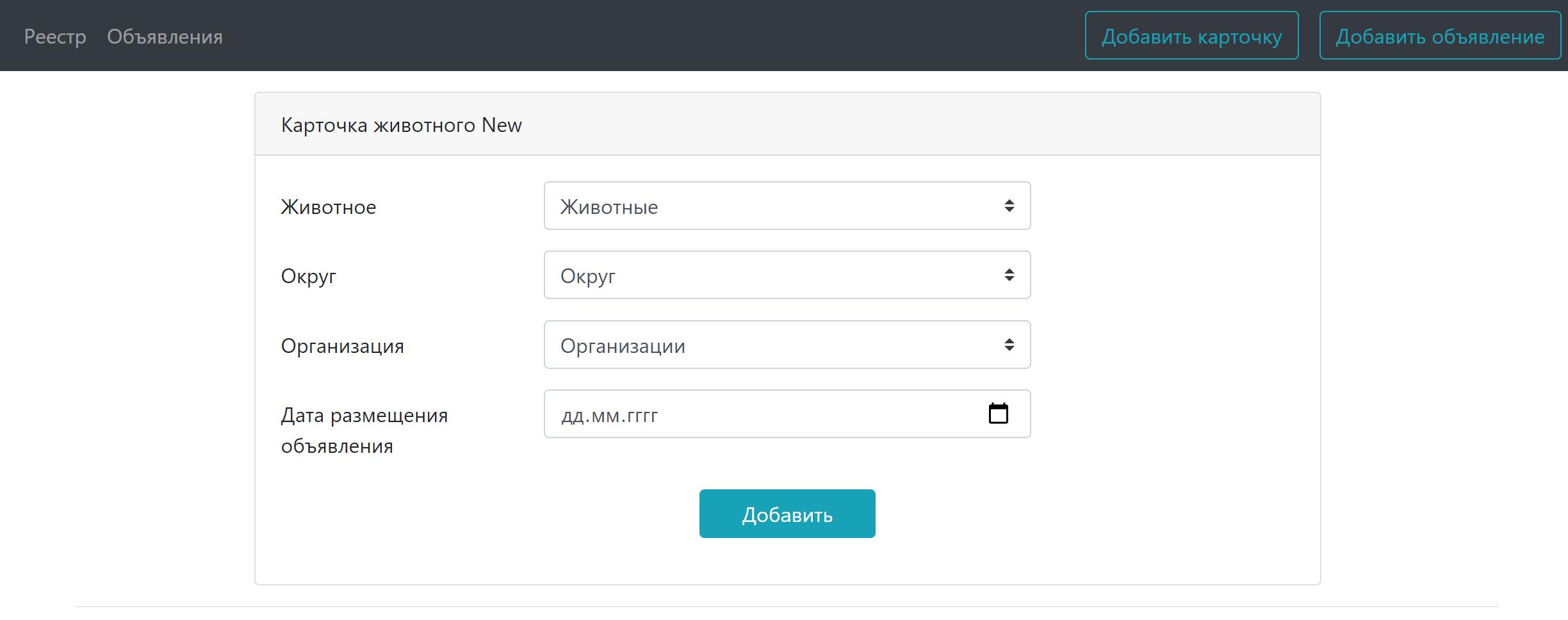


Рисунок 10. Страница для добавления карточки животного

## Инструментарий разработки

Так как в качестве основного языка для ПО был выбран C#, поэтому программной средой разработки стала Visual Studio. Программа была создана на основе веб-технологий ASP.NET MVC Framework. Для внешнего оформления страниц была использована css-библиотека Bootstrap.

# ЗАКЛЮЧЕНИЕ

В ходе выполнения данной работы были выполнены следующие этапы:

1. определены требования к системе через пользовательские прецеденты;
2. определен набор концептов с помощью выделения концептуальных классов;
3. определены последовательности действий для прецедентов через системные диаграммы последовательностей;
4. установлены взаимодействия между компонентами с помощью диаграмм коммуникаций;
5. создана общая иерархия классов с помощью диаграммы классов проектирования;
6. создана схема данных;
7. создано программное обеспечение с реестром для пользователей всех организаций и округов.

Полученное ПО осуществляет:

1. возможность просматривать записи в различных представлениях,
2. возможность фильтровать и сортировать внутренний реестр по характеристикам животных,
3. возможность добавлять новые карточки в реестр,
4. для администратора возможность удалять записи реестра,
5. открытие определенных частей реестра для пользователей с разными ролями.

Данное ПО легко в использовании, а использованный шаблон MVC позволяет масштабировать и улучшать программный продукт.

Исходный код можно посмотреть на платформе GitHub [2].

# СПИСОК ЛИТЕРАТУРЫ И ССЫЛОК

1. К. Ларман «Проектирование UML и шаблонов проектирования». – Второе издание, - Москва: Вильямс, 2004, - 609 c.
2. Animaux, проект на GitHub. URL: <https://github.com/ISMAKOVA/PIS_ANIMAUX>.