**Министерство образования Иркутской области**

Государственное бюджетное профессиональное

образовательное учреждение Иркутской области

«Иркутский авиационный техникум»

(ГБПОУИО «ИАТ»)

КР.09.02.07-5.24.211.23 ПЗ

ПРИЛОЖЕНИЕ

«ВЕТЕРИНАРНАЯ КЛИНИКА»

|  |  |  |
| --- | --- | --- |
| Председатель ВЦК: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (подпись, дата) | (М.А. Кудрявцева) |
| Руководитель: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (подпись, дата) | (Н. Р. Карпова) |
| Студент: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (подпись, дата) | (Л.С. Чиин) |

Иркутск 2024

**Содержание**

[Введение 4](#_Toc165409120)

[1. Описание предметной области 4](#_Toc165409121)

[2. Анализ инструментальных средств разработки 4](#_Toc165409122)

[3. Техническое задание 4](#_Toc165409123)

[4. Проектирование ПП 4](#_Toc165409124)

[4.1. Структурная схема ПП 4](#_Toc165409125)

[4.2 Функциональная схема ПП 4](#_Toc165409126)

[4.3 Проектирование базы данных 4](#_Toc165409127)

[4.4 Проектирование интерфейса 4](#_Toc165409128)

[5. Разработка приложения 4](#_Toc165409129)

[5.1 Разработка интерфейса программы 4](#_Toc165409130)

[5.2 Разработка базы данных ПП 4](#_Toc165409131)

[5.3 Разработка программы 4](#_Toc165409132)

[6. Документирование программного продукта 4](#_Toc165409133)

[6.1 Руководство пользователя ПП 4](#_Toc165409134)

[Заключение 4](#_Toc165409135)

[Список используемых материалов 4](#_Toc165409136)

[Приложение А – Техническое задание 4](#_Toc165409137)

[Приложение Б – Листинг FormLogin.cs 4](#_Toc165409150)

Введение

Почти в жизни каждого человека появлялось желание завести домашнего питомца, не будь то кошка, собака или морская свинка. И по статистике уже у многих он есть, за время жизни с ними люди понимают, что животные болеют так же, как и мы, и что их надо лечить. Из-за этого сейчас большой спрос на ветеринарные клиники. Каждая из них пытается предложить более хорошие условия: качественный сервис, низкие цены, быстрое время обслуживания и так далее. Удобное приложение тоже является не маловажным фактором, ведь мы живём в век технологий и люди обращают на это внимание на ровне с другими факторами.

С помощью результата курсовой работы взаимодействие с ветеринарной клиникой станет удобнее, приложение предоставит возможность просмотра информации о клинике, услугах, и о её ветеринарах, так же будет возможность записи на приём c возможностью выбора времени и ветеринара. В приложении будет реализована система регистрации и авторизации. В личном кабинете у пользователя будет возможность изменить контактную информацию и информацию о своих питомцах, будет возможность добавления сразу нескольких животных.

Цель: разработка программного продукта – приложение «Ветеринарная поликлиника», с удобным интерфейсом.

Для осуществления обозначенной цели служат следующие задачи:

1. Провести предпроектное исследование.
2. Разработать техническое задание.
3. Разработка алгоритма системы.
4. Разработка клиентской части продукта, Frontend.
5. Разработка программно-аппаратную части продукта, Backend.
6. Разработка базы данных.
7. Описание предметной области

Предметной областью курсовой работы является ветеринарная клиника.

Ветеринарная клиника − лечебно-профилактическое учреждение для оказания ветеринарной помощи больным животным на приёме в специализированном учреждении, либо на дому. Различают государственные и частные ветеринарные клиники, но государственных очень мало. У каждой клиники есть штат работников и клиентов. Сейчас очень много учреждений такого типа, из-за чего в этой сфере высокая конкуренция, для завлечь людей нужно открывать филиалы в удобных местах, предоставлять качественные услуги, ну и, конечно, средства рекламы.

При составлении графика работы учитывается количества работников, размер помещения, большинство клиник работают ежедневно, до 8-10 часов вечера. У ветеринаров есть своя специализация, кто-то работает с домашними животными по типу кошек и собак, другие с более экзотическими животными, например кролики и морские свинки.

Ветеринарная клиника имеет различное оборудование и инструменты для проведения диагностики и лечения животных. К ним относятся лабораторное оборудование, рентгеновские аппараты, аппараты для проведения ультразвуковых исследований, операционные столы, стерилизаторы и другое.

Клиника может предоставлять такие услуги как диагностику заболеваний, проведение операций, предоставление консультаций, назначение и осуществление лечения, вакцинации, контроль за состоянием животного, а также груминг. Не редко учреждение может продавать ветеринарные препараты и корма.

Объектами данной предметной области выступают администратор, ветеринары и клиенты.

Администратор обладает правом доступа к базе данных для изменения информационного наполнения. К его компетенции относится разработка и согласование графика работы специалистов, обновление актуальной информации о клинике, а также редактирование ассортимента предоставляемых услуг, в том числе их наименования, стоимости и описания.

Ветеринар – это специалист в области ветеринарной медицины, который осуществляет практическую деятельность согласно стандартам профессии. Его функционал включает в себя доступ к приложению, которое позволяет просматривать данные о клиентах, их домашних животных, а также расписание приемов.

Пациент – это клиент ветеринарной клиники, который обращается за медицинской помощью для своих домашних животных. Ему предоставляется доступ к информационным ресурсам клиники, включая сведения о медицинском персонале, расписании работы и ассортименте предоставляемых ветеринарных услуг. При посещении клиники пациент может получить необходимую консультацию, диагностику и лечение в рамках установленных стандартов профессиональной практики.

В соответствии с предметной областью можно выделить взаимодействующие

сущности и их атрибуты:

Пользователь. Атрибуты – Код пользователя, Логин, Пароль, Роль, Имя, Фамилия, Отчество.

Питомец. Атрибуты – Код питомца, Имя, Дата рождения, Код хозяина, Болезни.

Услуга. Атрибуты – Код услуги, Название, Описание.

Информация. Атрибуты – Код информации, Заголовок, Содержание.

Запись. Атрибуты – Код записи, Код услуги, Код питомца, Код пользователя, Дата, Время, ФИО ветеринара.

Сущности взаимодействуют между собой, создавая полномерный процесс работы:

1. Администратор:

Имеет доступ к содержимому всех таблиц из базы данных, кроме одной - role.

Редактирует список приёмов, проверяет корректность заполненных данных в разных таблицах.

1. Клиент:

Добавляет и редактирует информацию о своём аккаунте, а также добавляет и редактирует информацию о своих питомцах в базе данных клиники.

Просматривает информацию о клинике и её услугах.

Начинает запись на приём, выбирает удобное время из доступного, выбирает ветеринара и подтверждает запись.

1. Ветеринар:

Просмотр данные клиентов и их питомцев из базы данных клиники.

Просматривает расписание приёмов и список всех услуг.

1. Приложение:

Обеспечивает доступ к информации о поликлинике.

Предоставляет возможность выбора услуги и записи на приём.

Предоставляет список приёмов и информацию о клиентах ветеринарам.

# **Анализ инструментальных средств разработки**

Проектировать структуру программного продукта можно с помощью MySQL Workbench и Draw.io или Microsoft Visio. Программный продукт будет состоять из двух частей: программно-аппаратная и клиентская.

Draw.io – онлайн-сервис, с помощью которого можно создавать блок-схемы, прототипы, инфографику и диаграммы любого типа. Выбор данного сервиса был определён следующими данными факторами: бесплатное использование, взаимодействие с продуктом через браузер, широкий функционал, который позволяет спроектировать алгоритм работы будущего программного продукта.

Microsoft Visio – векторный графический редактор, редактор диаграмм и блок-схем для Windows. Полнофункциональная версия Microsoft Visio Professional для создания и редактирования монограмм и диаграмм.

Самыми популярными языками программирования являются: Python, C, C++, C# и Java.

Python – это интерпретируемый, высокоуровневый язык программирования, который обладает простым синтаксисом и акцентирует внимание на читабельности кода. Он был создан Гвидо ван Россумом и изначально выпущен в 1991 году. Python широко используется в различных областях, включая веб-разработку, научные вычисления, искусственный интеллект и анализ данных. Он имеет обширную библиотеку стандартных модулей и поддерживает различные сторонние библиотеки, что делает его одним из самых популярных языков программирования.

C – это язык программирования, созданный в 1972 году в форме расширения языка программирования B. Он является компилируемым языком и обеспечивает низкоуровневый доступ к памяти и аппаратным средствам компьютера. C широко используется для разработки операционных систем, встроенных систем и производительного программного обеспечения. Он также является основой для различных языков программирования высокого уровня, таких как C++ и C#.

C++ – это компилируемый язык программирования, разработанный как расширение языка C. Он предоставляет возможности объектно-ориентированного программирования, что позволяет разработчикам организовывать код в логические сущности, называемые классами. C++ также поддерживает множество других функций, таких как перегрузка операторов, шаблоны и исключения. Он широко используется для разработки приложений и игр, а также для создания высокопроизводительных системного программного обеспечения.

C# – это объектно-ориентированный язык программирования, разработанный компанией Microsoft. Он является частью платформы .NET и основывается на синтаксисе языков программирования C++ и Java. C# предоставляет разработчикам широкий спектр возможностей, включая исключения, делегаты, события, атрибуты и LINQ. Он используется для разработки разнообразных приложений, в том числе десктопных, веб-приложений, игр и мобильных приложений.

Java – это объектно-ориентированный язык программирования, разработанный компанией Sun Microsystems (ныне принадлежит Oracle). Он является компилируемым языком и также работает на виртуальной машине Java (JVM). Java обеспечивает множество функций, включая автоматическую память сборщика мусора, многопоточность и возможность обработки исключений. Он широко используется для создания веб-приложений, мобильных приложений, больших корпоративных систем и игр. Java также является одним из самых популярных языков программирования благодаря своей платформенной независимости и обширной экосистеме библиотек и фреймворков.

В качестве языка программирования при создании программного продукта будет использоваться Python, по ряду следующих причин:

* Кроссплатформенность, программа написанная на python может работать на различных системах, таких как Windows, MasOS, Android. Так же на Python можно создавать веб-приложения.
* Мощный и широкий выбор библиотек: Python имеет большое количество библиотек и фреймворков.
* Простота в изучении: Python имеет простой и понятный синтаксис, который легко освоить.
* Большое сообщество разработчиков: Python имеет активное сообщество разработчиков, готовых помочь и поделиться решениями проблем и развивать данный язык.

Для программирования на Python можно рассмотреть 2 среды разработки: PyCharm и Microsoft Visual Studio.

PyCharm – это интегрированная среда разработки для языка программирования Python. Он предоставляет мощные инструменты и функции, которые помогают разработчикам создавать, отлаживать и поддерживать программы на Python. PyCharm имеет широкий набор возможностей, включая авто-дополнение кода, интеграцию с системой управления версиями, интегрированный терминал, возможность создания виртуальных окружений, отладчик, тестирование кода, анализ кода и другие функциональности, которые повышают производительность и удобство программирования на Python. PyCharm также обладает понятным и интуитивно понятным пользовательским интерфейсом, что делает его доступным даже для новичков в программировании.

Microsoft Visual Studio – это интегрированная среда разработки, разработанная компанией Microsoft, которая широко используется разработчиками для создания различных типов программного обеспечения, включая веб-приложения, настольные приложения, мобильные приложения, облачные сервисы, игры и многое другое. Поддерживает большое количество языков программирования, таких как C#, C++, Visual Basic, Python и других.

Для создание своего программного продукта будет использоваться – PyCharm, а именно его версия PyCharm Community Edition. Обосновывается этот выбор тем, что данная среда разработки полностью бесплатна, удобна и проста в использовании и содержит все нужные функции. Хоть и Microsoft Visual Studio поддерживает больше языков программирования, но использоваться при создании программного продукта они не будут.

Таблица 1 – Сравнение сред разработки

|  |  |  |
| --- | --- | --- |
| Параметры | Microsoft Visual Studio | PyCharm |
| Распространяется бесплатно | + | + |
| Автоматическое сохранение | + | + |
| Подсказки по коду | + | + |
| Интеграция с системой контроля версия (GIT) | + | + |
| Заточен под Python-разработку | - | + |
| Поддержка CSS/HTML/JS | + | + |

В качестве СУБД будет использоваться MySQL Workbench. MySQL Workbench – это система управления базами данных (СУБД), которая предоставляет надежное и эффективное хранение, управление и доступ к структурированным данным. Она основана на языке SQL и позволяет создавать, изменять и удалять базы данных, таблицы, индексы и другие объекты базы данных. MySQL поддерживает множество функций, таких как многопользовательский доступ, транзакции, сохранение целостности данных и взаимодействие с различными языками программирования. Она широко используется веб-приложениями и веб-сайтами, а также является одной из самых популярных СУБД в мире.

1. Техническое задание

В начале разработки создавалось техническое задание, в котором указывались основные требования.

Для создания технического задания использовался стандарт ГОСТ 34.602-2020.

Согласно ГОСТ 34.602-2020 техническое задание должно включать следующие разделы:

1. общие сведения;
2. назначение и цели создания ПП;
3. требования к системе в целом;
   1. требования к структуре и функционированию ПП;
   2. требования к надежности;
   3. требования к безопасности;
   4. требования к эксплуатации, техническому обслуживанию, ремонту и хранению компонентов ПП;
4. требования к документированию;
5. состав и содержание работ по созданию ПП.

Техническое задание на разработку приложения представлено в приложении А.

1. Проектирование ПП
   1. Структурная схема ПП

Проектирование приложения для ветеринарной клиники, началось построения диаграммы использования. Диаграмма прецедентов Uses CASE представлена на рисунке 1. Она содержит 3 актёра: пользователь, ветеринар, администратор. Каждый выполняет свои определённые функции, всего функций 10.

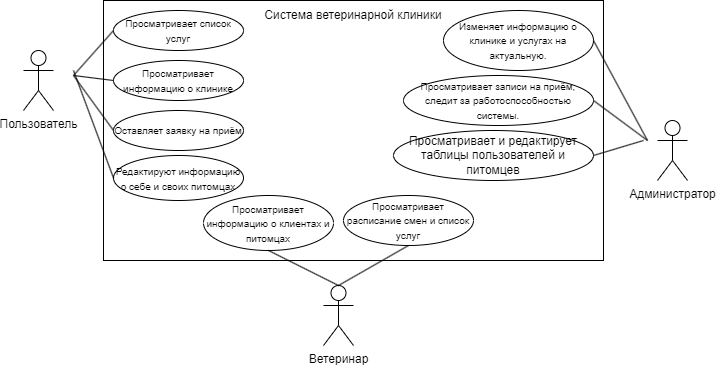


Рисунок 1 − Диаграмма прецедентов

На следящем рисунке под номером 2 представлена диаграмма деятельности. Диаграмма содержит 2 роли: база данных и пользователь. Процесс начинается с пользователя, у которого есть четыре варианта действий: просмотр услуг, просмотр информации о клинике и запись на приём. После просмотра информации или услуг, а также добавления питомцев пользователь может выбрать любое из четырёх действий. После выбора записи на прём происходит цыпочка действий с выбором нужного времени, ветеринара, а также услуги. Если пользователя всё устраивает, то он подтверждает запись и она добавляется в базу данных. База данных в этих процессах предоставляет и получает различные данные.

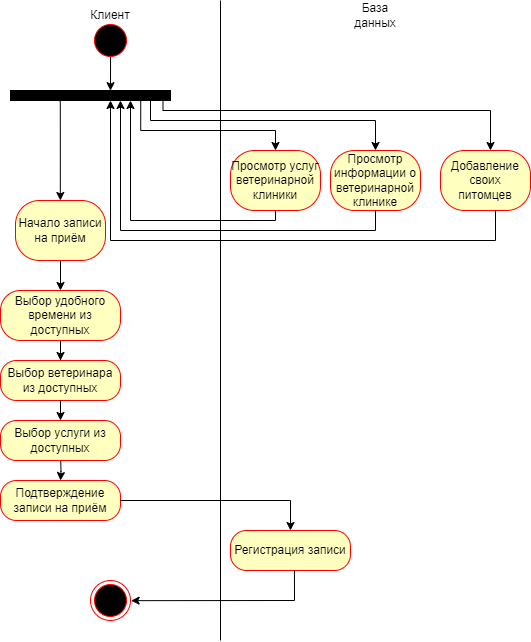


Рисунок 2 - Диаграмма деятельности

* 1. Функциональная схема ПП

На рисунке 3 представлена контекстная диаграмма, отображающая деятельность ПП.

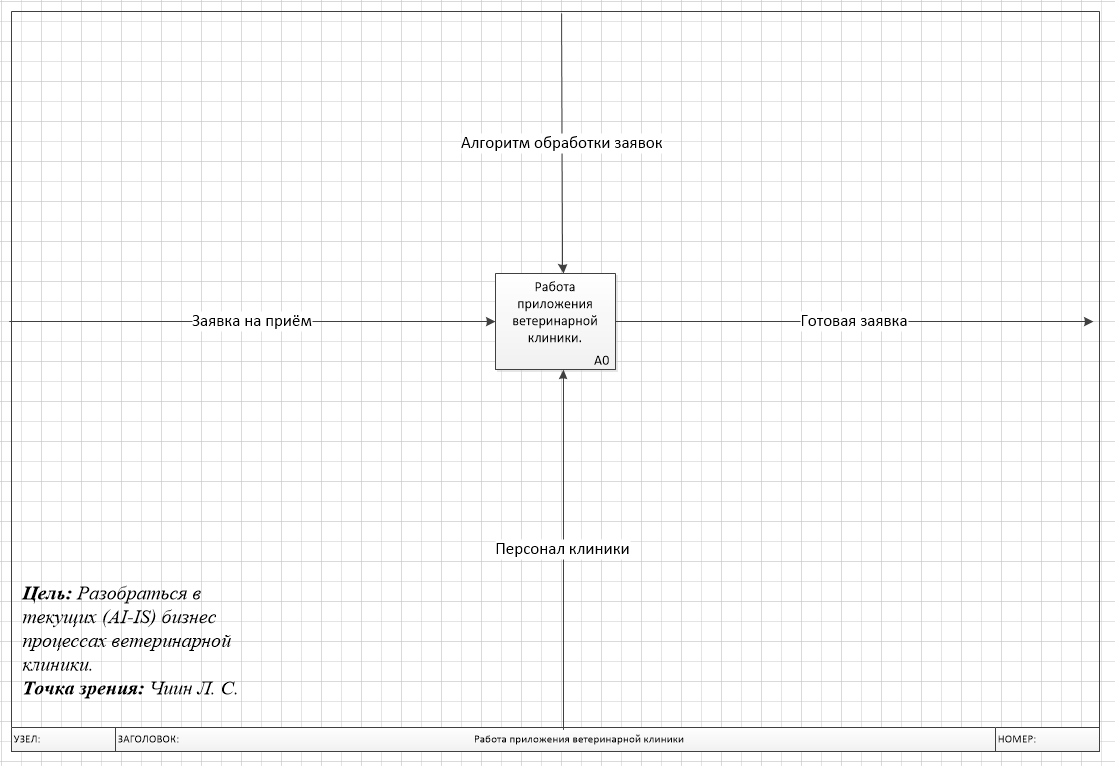


Рисунок 3 - Контекстная диаграмма IDEF0

На следящем рисунке под номером 4 представлена диаграмма декомпозиций (А1), отображающая деятельность ПП более подробно предыдущей.

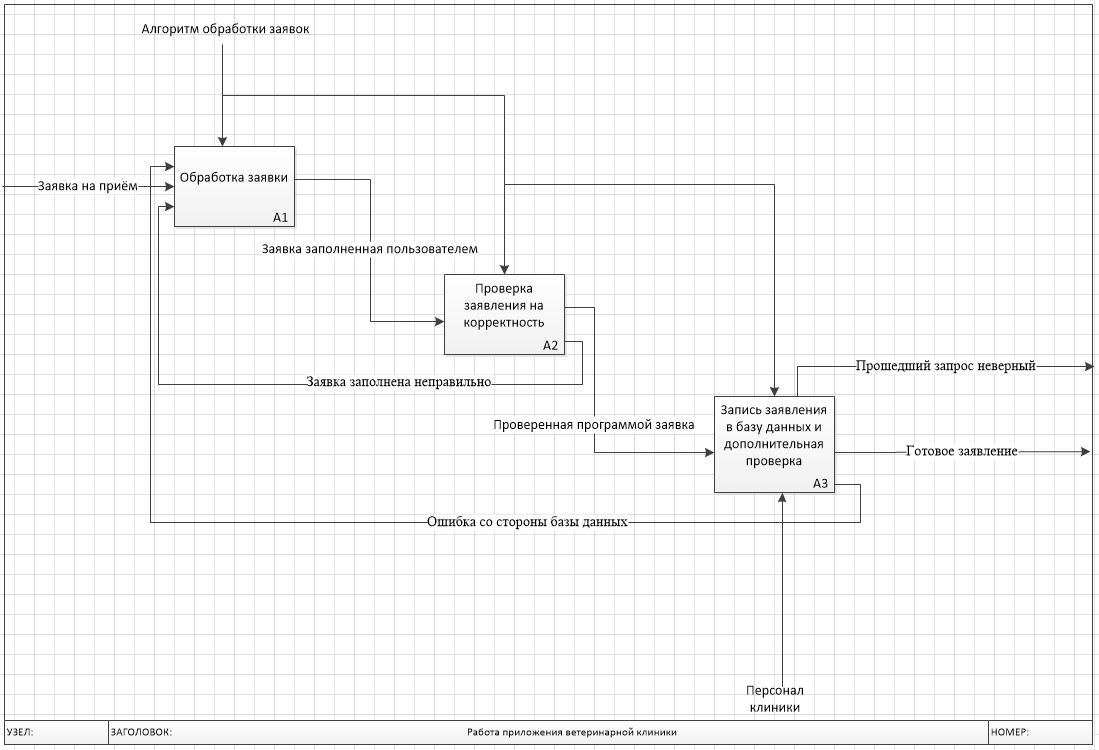


Рисунок 4 − Диаграмма декомпозиций (А1)

* 1. Проектирование базы данных

Первый этап создания базы данных – это разработка ее концептуальной модели.

Концептуальное проектирование – это этап процесса проектирования, на котором осуществляется разработка абстрактного общего плана или концепции будущего продукта, системы или услуги. На этом этапе проектировщики определяют основные принципы, цели, требования и ограничения проекта, а также создают концептуальную модель будущего продукта или системы.

Инфологическая модель базы данных представлена на рисунке 5.

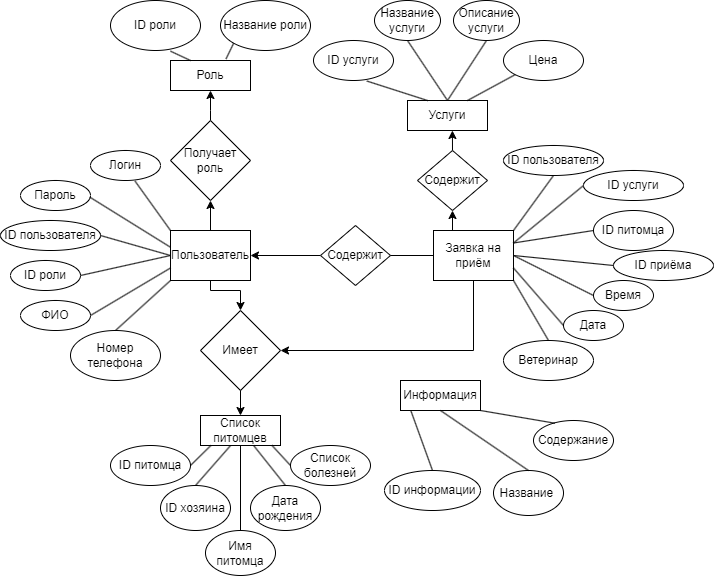


Рисунок 5 − Инфологическая модель базы данных

На инфологической модели базы данных схематично отображены сущности системы, их атрибуты и связи между ними. В прямоугольниках отображены различные сущности, всего их 6: информация, список питомцев, заявка на приём, пользователь, услуги и роли. В кругах и так же овалах о отображены атрибуты этих сущностей, например: ID питомца, ID роли, ФИО и так далее. Связи изображены ромбами.

Затем происходит преобразование концептуальной модели в логическую, с использованием формальных правил. Таким образом, логическое проектирование подразумевает создание схемы базы данных на основе определенной модели данных, например реляционной. Основное внимание уделяется описанию структуры данных и их взаимосвязей, чтобы обеспечить эффективное функционирование базы данных и удовлетворить основные потребности пользователей.

Рисунок 6 показывает даталогическую модель базы данных, на ней показаны сущности приложения, а также первичные ключи и внешние ключи, демонстрирующие связи сущностей между собой.

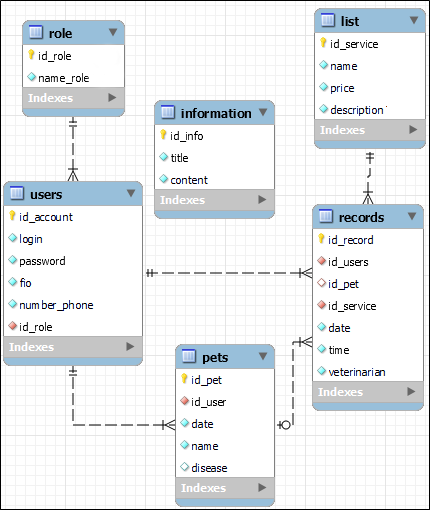


Рисунок 6 − Даталогическая модель базы данных

На следящем рисунке под номером 7 изображена ER-диаграмма базы данных. Для её полного функционирования и качественной работоспособности понадобилось N таблиц.

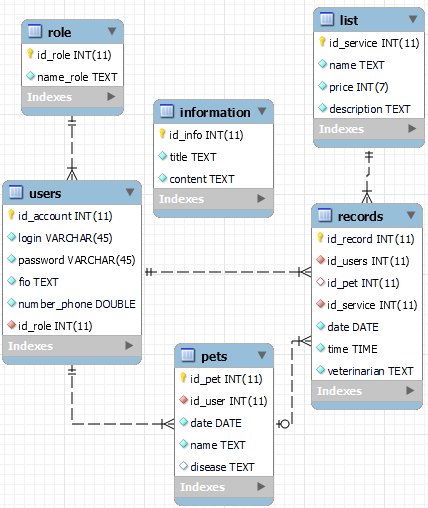


Рисунок 7 – ER-модель базы данных

Перечень таблиц представлена в таблице 2. Поля, типы данных и описание столбцов каждой таблицы представлены в таблицах с 3 по 8.

Таблица 2 – Таблицы ER-модели

|  |  |
| --- | --- |
| Таблица | Описание |
| role | Таблица, которая содержит роли |
| users | Таблица, которая содержит информацию о пользователя, администраторах, ветеринарах. |
| pets | Таблица, которая содержит информацию о домашних животных клиентов. |
| records | Таблица, которая содержит информацию о записях. |
| list | Таблица, которая содержит информацию о списке услуг. |
| information | Таблица, которая содержит общую информацию о клинике, например, адрес и контакты. |

Таблица 3 – Таблица «users»

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id\_account | INT(11) | Код пользователя |
| login | VARCHAR(45) | Логин |
| password | VARCHAR(45) | Пароль |
| fio | TEXT | ФИО |
| number\_phone | DOUBLE | Номер телефона |
| id\_role | INT(11) | Код роли |

Таблица 4 – Таблица «role»

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id\_role | INT(11) | Код роли |
| name\_role | TEXT | Название роли |

Таблица 5 – Таблица «pets»

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id\_pet | INT(11) | Код питомца |
| id\_user | INT(11) | Код пользователя |
| date | DATE | Дата рождения |
| name | TEXT | Имя |
| disease | TEXT | Медицинская справка, список болезний |

Таблица 6 – Таблица «information»

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id\_info | INT(11) | Код информации |
| title | TEXT | Заголовок |
| content | TEXT | Содержание |

Таблица 7 – Таблица «list»

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id\_service | INT(11) | Код услуги |
| name | TEXT | Название |
| price | INT(7) | Цена |
| description | TEXT | Описание |

Таблица 8 – Таблица «records»

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id\_record | INT(11) | Код записи |
| id\_users | INT(11) | Код пользователя |
| id\_pet | INT(11) | Код питомца |
| id\_service | INT(11) | Код услуги |
| date | DATE | Дата |
| time | TIME | Время |
| veterinarian | TEXT | ФИО ветеринара |

4.4 Проектирование интерфейса

Для разработки прототипа пользовательского интерфейса был выбран инструмент draw.io – это инструмент для создания блок-схем, диаграмм и прочего.

Были спроектированы три прототипа будущих страниц приложения: страница авторизации (рисунок 8), страница регистрации (рисунок 9), а также панель администратора (рисунок 10). Создание прототипов будущих страниц помогает и ускоряет процесс разработки программного продукта.

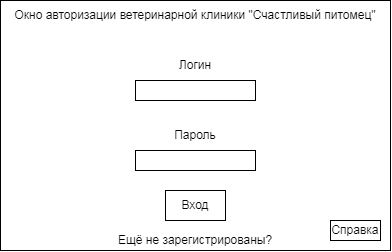


Рисунок 8 – Страница авторизации

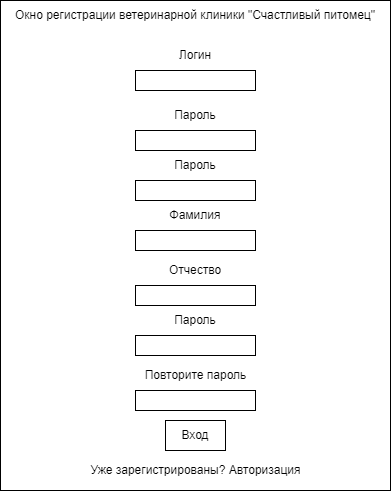


Рисунок 9 – Страница регистрации

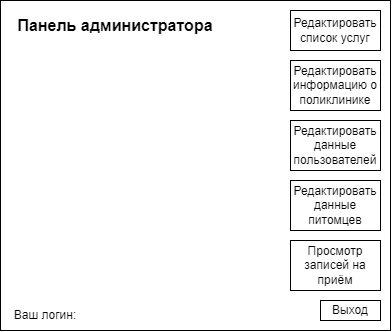


Рисунок 10 – Панель администратора

5. Разработка приложения

5.1 Разработка интерфейса программы

В приложении «Ветеринарная клиника «Счастливый питомец»» были реализованы следующие функции:

* Регистрация;
* Авторизация по ролям;
* Просмотр такой информации о клинике как: номер телефона, адрес и график работы;
* Изменение данных в личном кабинете, для пользователя;
* Запись на приём с выбором даты, времени, услуги, ветеринара и питомца, для пользователя;
* Просмотр списка своих записей на приём, для пользователя;
* Добавление питомцев с возможностью выбора даты рождения, имени и списка болезней, для пользователя;
* Просмотр списка услуг и информации о них для пользователя и ветеринара;
* Просмотр такой информации как ID, ФИО и номер телефона о всех пользователях приложения, для ветеринара;
* Просмотр информации о питомцах, для ветеринара;
* Просмотр списка всех записей на приём, для ветеринара;
* Просмотр и редактирование списка услуг, для администратора;
* Просмотр и редактирование информации о клинике, для администратора;
* Просмотр и редактирование списка пользователей, для администратора;
* Просмотр и редактирование списка питомцев, для администратора;
* Просмотр и редактирование списка записей на приём, для администратора.

Интерфейс приложения понятен, прост и интуитивен, разобраться в нём не трудно.

Объём exe файла и библиотеки нужной для запуска составляет 2,65 МБ, рисунок 11.



Рисунок 11 – Объём exe файла и библиотеки

Объём потребляемой ОЗУ составляет 10,4 МБ, рисунок 12.



Рисунок 12 – Объём потребляемой ОЗУ

Все поля ввода в приложении проверяют вводимы данные на корректность, и в случае ошибки предупреждают пользователя.

Приложение будет работать на любом устройстве который соответствует системным требованиям и обладает версией Windows 7 и более.

Разработка приложения началась с создания системы регистрации и авторизации. Для каждой функции будет создан отдельный класс и отдельная форма. Класс для системы авторизации будет называться Authorization и его код будет представлен на рисунке 13, форма FormLogin будет представлена на рисунке 14, а части кода будут представлены на рисунках 15 и 16. Для регистрации это будет класс Registration, рисунок 17 и форма FormRegister, рисунок 18, а код формы на рисунках 19 и 20.



Рисунок 13 – Класс Authorization

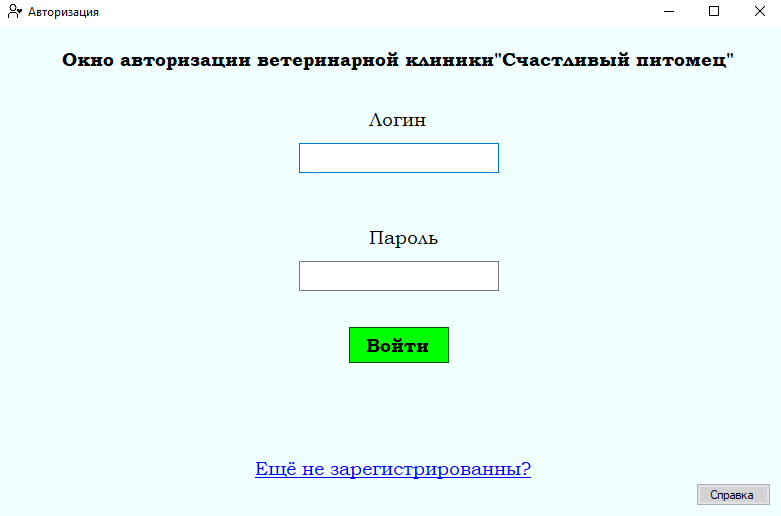


Рисунок 14 – Форма FormLogin

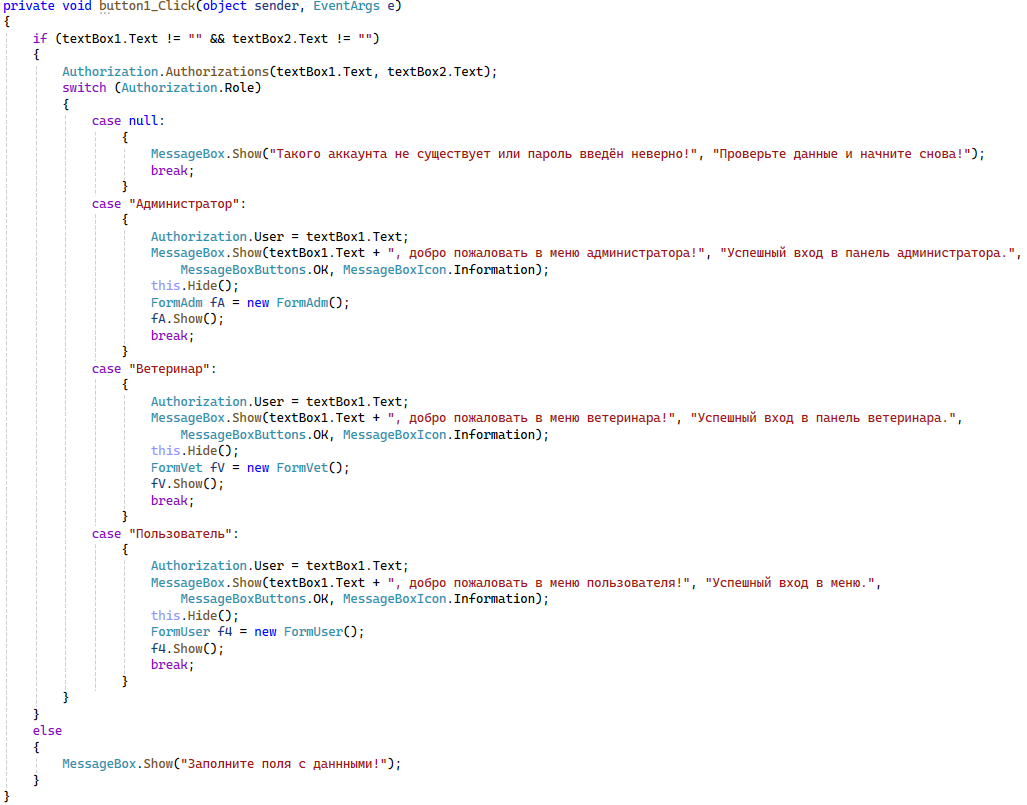


Рисунок 15 – Код кнопки «Вход» на FormLogin

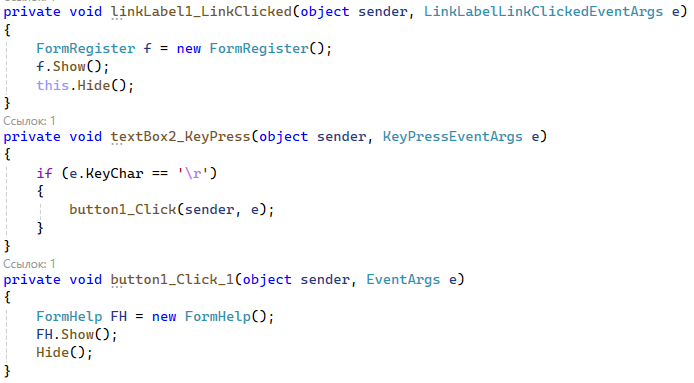


Рисунок 16 – Остальная часть кода FormLogin



Рисунок 17 – Класс Registration

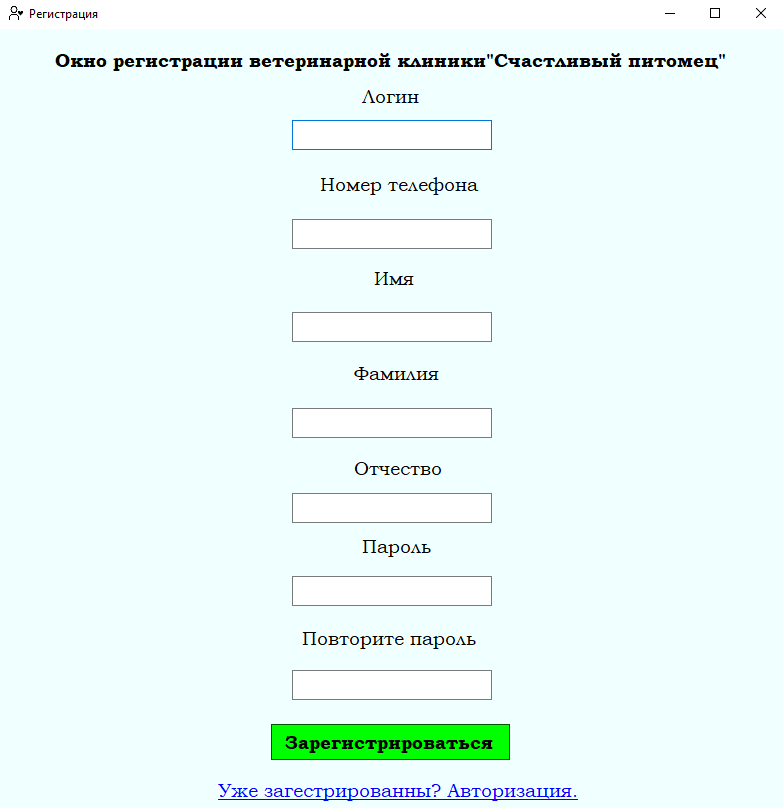


Рисунок 18 – Форма FormRegister



Рисунок 19 – Кнопка «Регистрация» на FormRegister

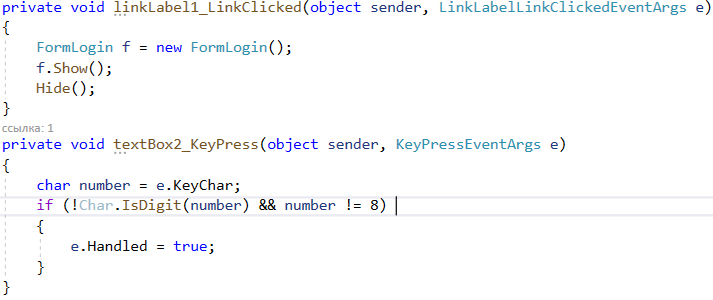


Рисунок 20 – Остальная часть кода FormRegister

На форме авторизации есть кнопка «Справка», которая пересылает нас на форму FormHelp, которая представлена на рисунке 21.

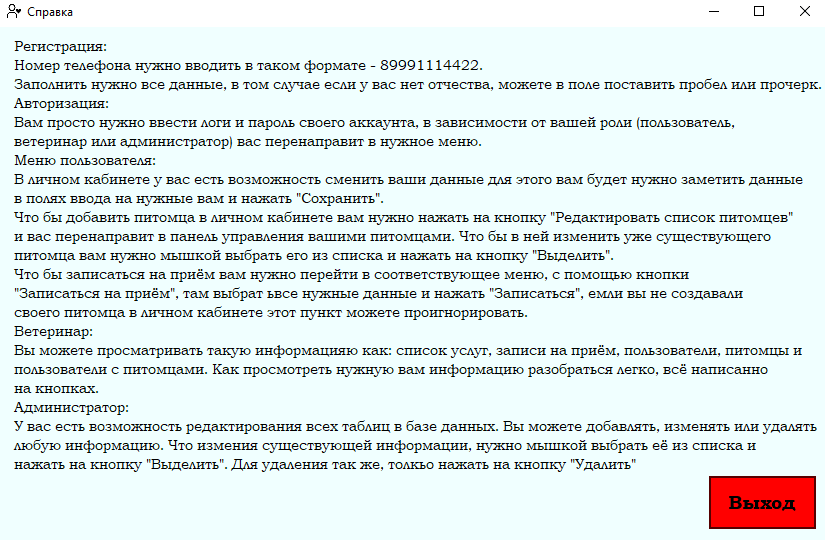


Рисунок 21 – Форма FormHelp

Дальше началась разработка панели пользователя, а именно её главное меню, на нём есть различная информация, такая как: логин пользователя, номер телефона клиники, график работы и адрес. Для получения информации используется класс Information, который отвечает за таблицу из базы данных с аналогичным названием и представлен на рисунках 22, 23 и 24. На его основе созданы остальные классы, которые отвечают за различные таблицы в базе данных. С левой части окна расположены четыре кнопки: записаться на приём, личный кабинет, список услуг и выход. Форма называется FormUser и представлена на рисунке 25, а её код на рисунке 26.



Рисунок 22 – Первая часть кода класса Information



Рисунок 23 – Вторая часть кода класса Information



Рисунок 24 – Третья часть кода класса Information

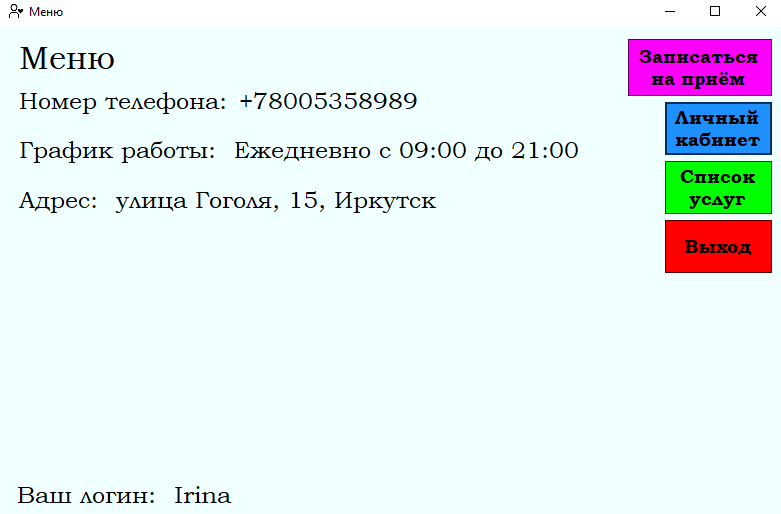


Рисунок 25 – Форма FormUser

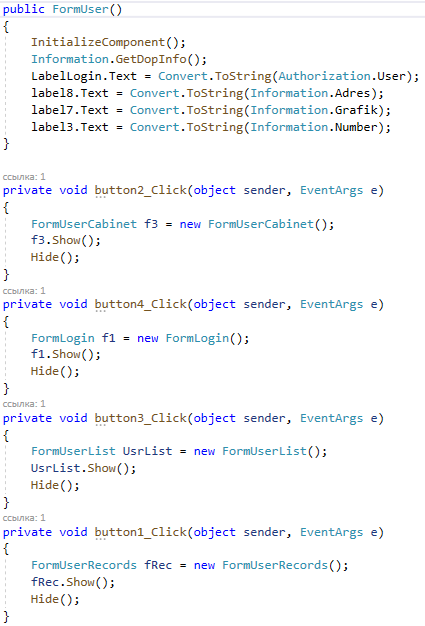


Рисунок 26 – Код формы FormUser

Главное меню в панели ветеринара выглядит практически аналогично с главным меню пользователя и тоже использует класс Information, но кнопки предоставляют другой функционал. Форма, отвечающая за главное меню ветеринара, называется FormVet, и представлена на рисунке 27, а её код на рисунке 28.

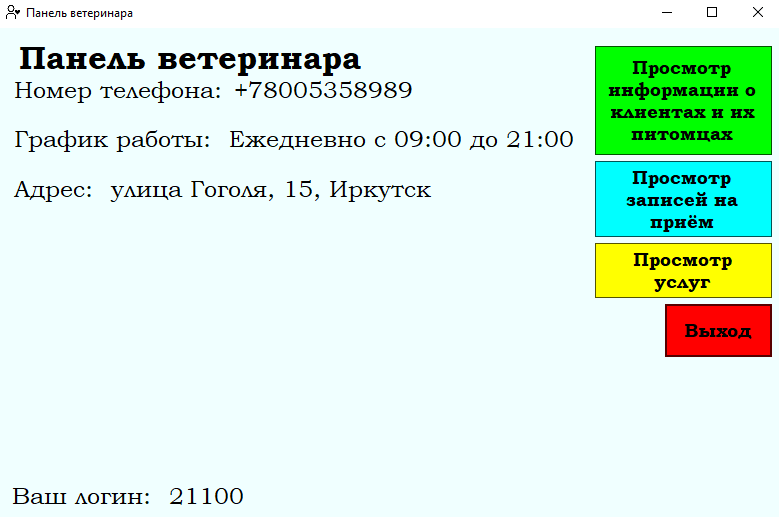


Рисунок 27 – Форма FormVet

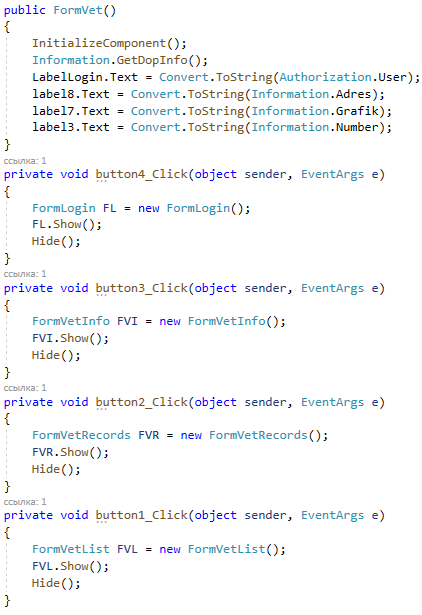


Рисунок 28 – Код формы FormVet

Последнее меню сделано для администратора, оно отличается тем, что в нём нету такой информации как: номер телефона клиники, график работы и адрес, и соответственно класс Information не используется. Сама форма называется FormAdm и представлена на рисунке 29, а её код на рисунке 30 и 31.

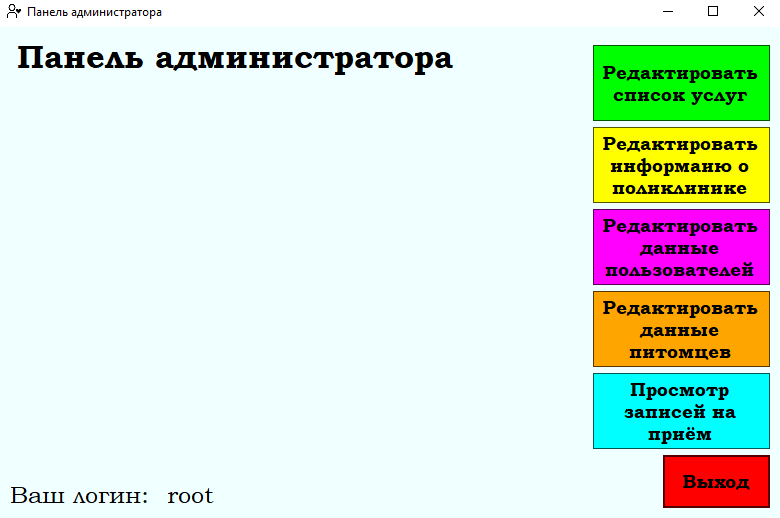


Рисунок 29 –Форма FormAdm

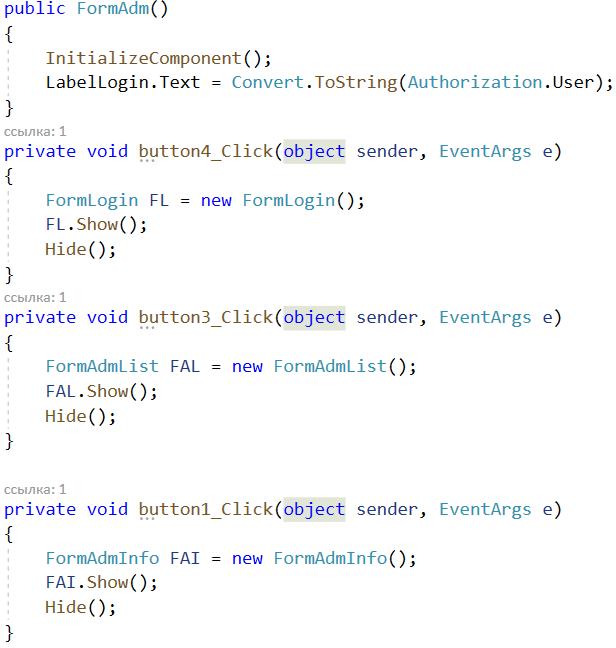


Рисунок 30 – Первая часть кода формы FormAdm

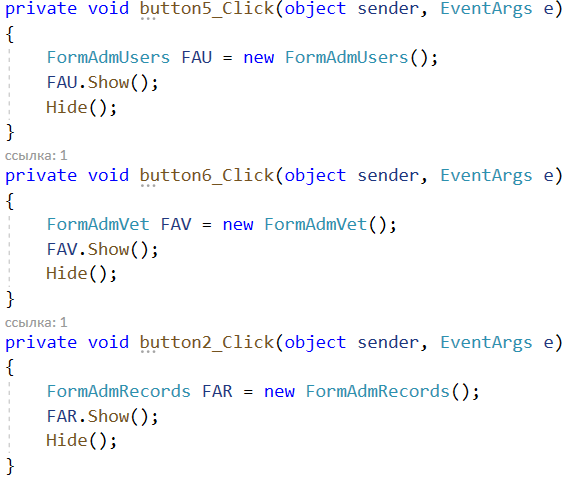


Рисунок 31 – Вторая часть кода формы FormAdm

В панели администратора предоставлены формы, отвечающие за таблицы в базе данных, для каждой таблицы своя форма. Все формы сделаны аналогично, только отвечают за разные таблицы и связанны с разными классами. Для примера будет продемонстрирована форма, отвечающая за таблицу information. Она связанна с классом Information который представлен на рисунках 22, 23 и 24. Форма называется FormAdmInfo и предоставляет возможность добавлять, удалять и редактировать данные в таблице information, представлена она на рисунке 32, а её код на рисунках 33, 34 и 35.

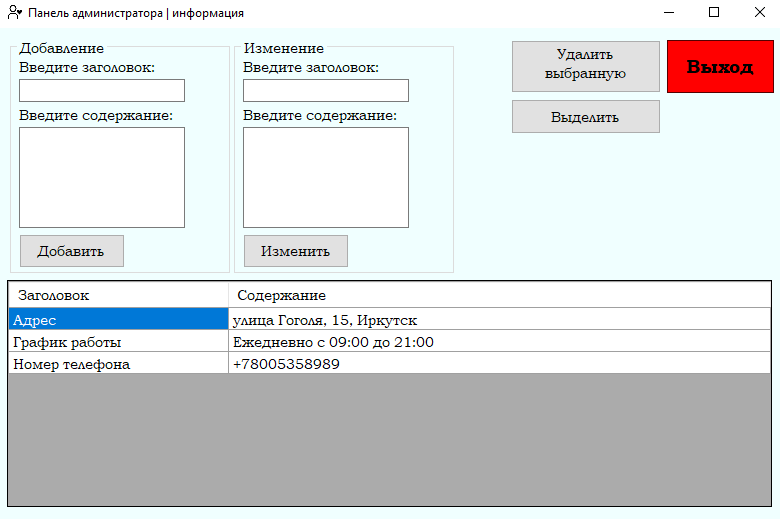


Рисунок 32 – Форма FormAdmInfo

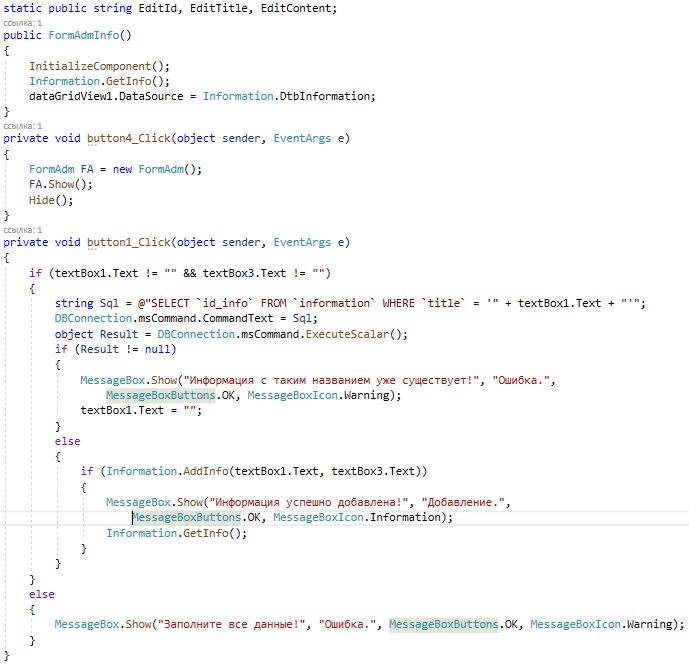


Рисунок 33 – Первая часть кода формы FormAdmInfo



Рисунок 34 – Вторая часть кода формы FormAdmInfo

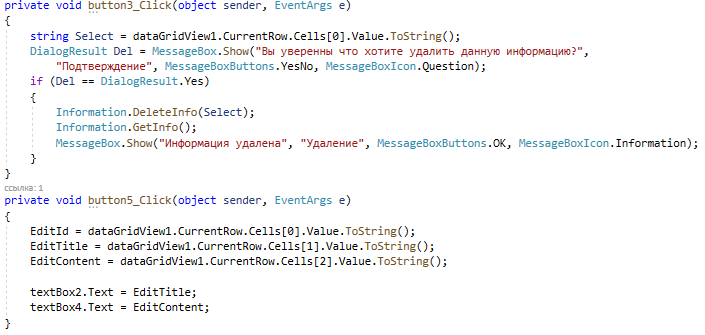


Рисунок 35 – Третья часть кода формы FormAdmInfo

Панель ветеринара содержит формы с просмотром различных таблиц из базы данных, таких как: pets, records, list и users, но с ограничениями, ветеринар не видит пароль и логин. Для records и list есть отдельные формы, а pets и users просматриваются в одной, называется эта форма FormVetInfo и представлена на рисунке 36, а её код на рисунке 37.

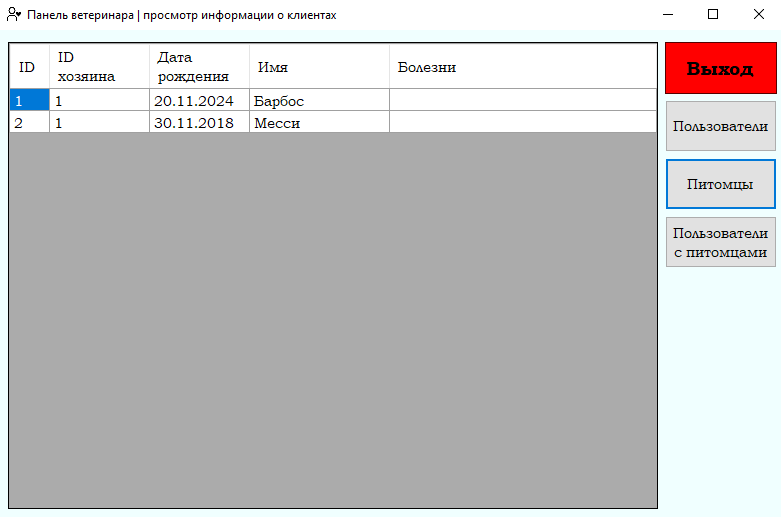


Рисунок 36 – Форма FormVetInfo

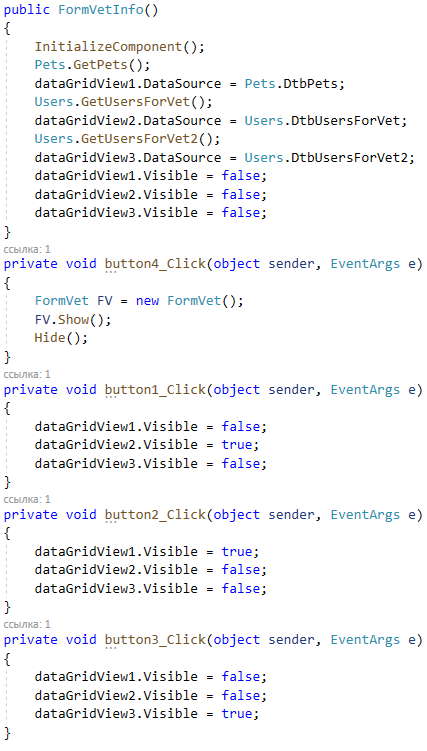


Рисунок 37 – Код формы FormVetInfo

У пользователя есть много форм, одна из них добавление питомцев, сделана она аналогично формам для администратора, только редактировать на ней можно только своих питомцев. Форма со списком услуг сделана так же, как у ветеринара, только имеет кнопку для перенаправления на форму с записью, и называется FormUserList, она представлена на рисунке 38.

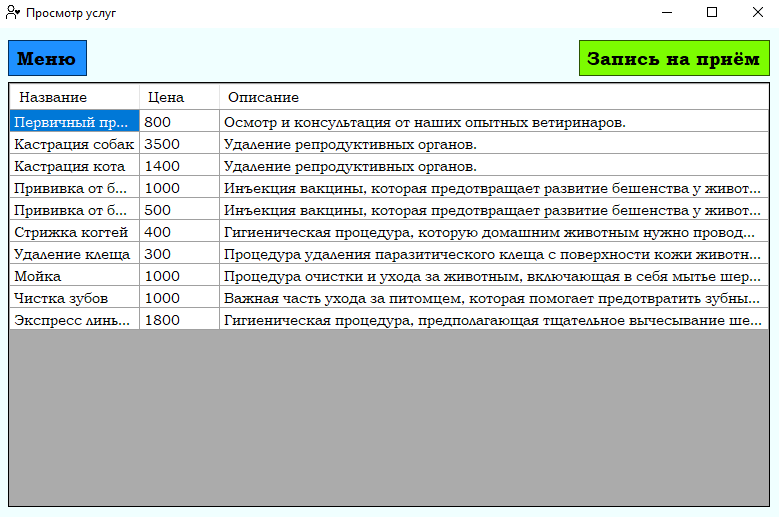


Рисунок 38 – Форма FormUserList

Личный кабинет позволяет перейти на форму с добавление питомцев или на форму, которая показывает список услуг. Сама форма личного кабинета называется FormUserCabinet, представлена на рисунке 39, а код кнопки сохранить на рисунке 40.

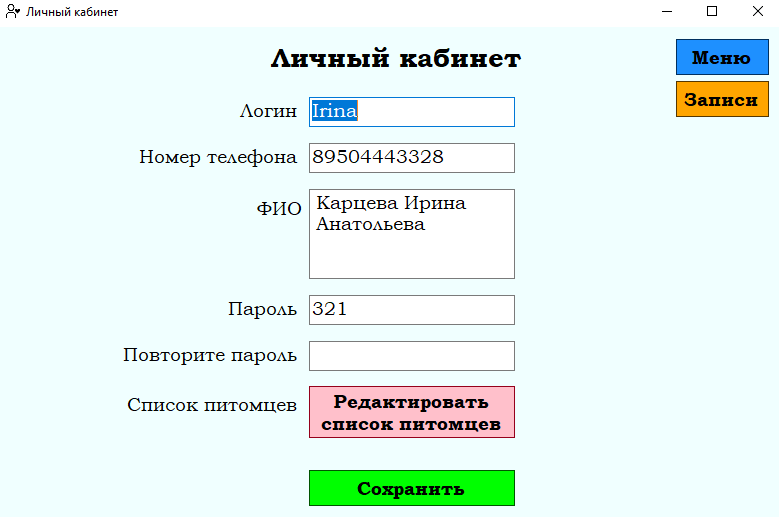


Рисунок 39 – Форма FormUserList

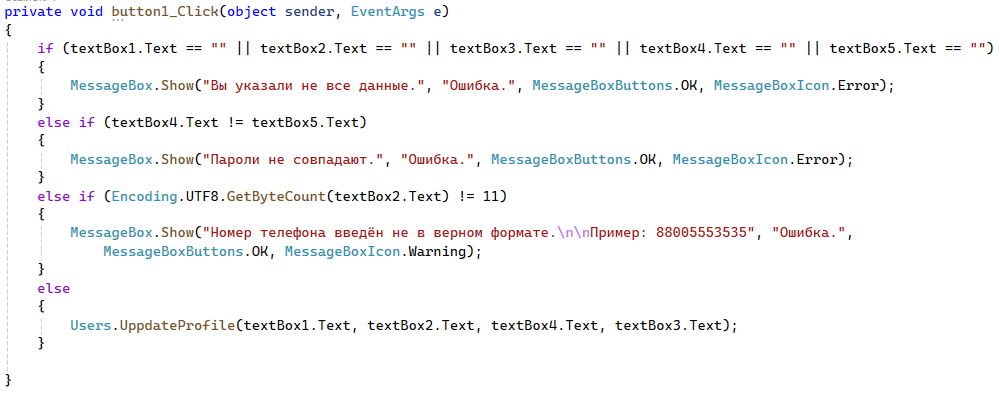


Рисунок 40 – Код кнопки «Сохранить» FormUserList

Форма FormUserRecList предоставляет возможность просматривать свои записи на приём с питомцем и без него отдельно, а также удалять их. Форма представлена на рисунке 41, а её код на рисунках 42 и 43.

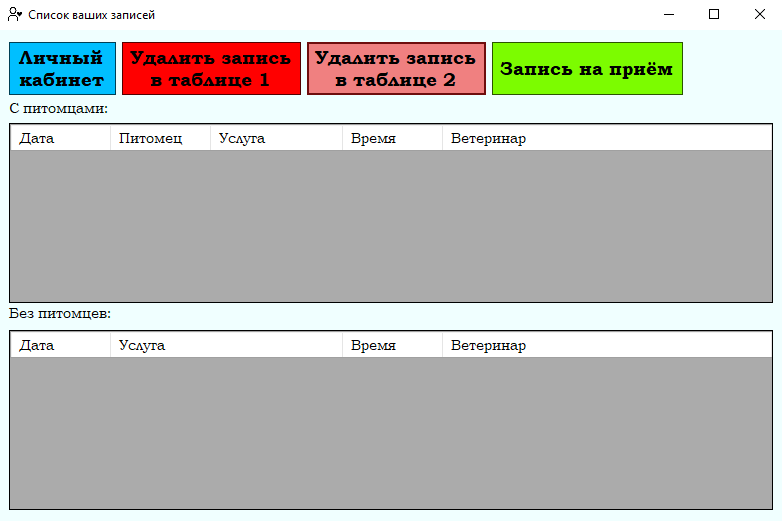


Рисунок 41 – Форма FormUserRecList

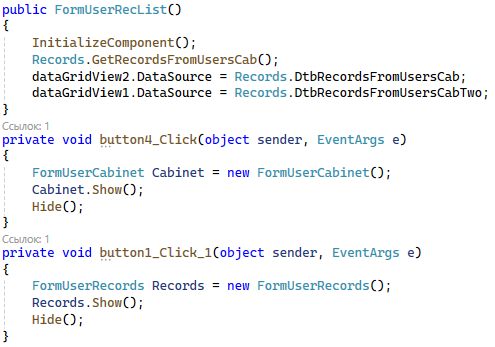


Рисунок 42 – Первая часть кода формы FormUserRecList



Рисунок 43 – Вторая часть кода формы FormUserRecList

Форма с записью на приём называется FormUserRecords и представлена на рисунке 44, на ней есть возможность выбрать все нужные параметры для записи на приём и записаться. Код данной формы представлен на рисунках 45 и 46.

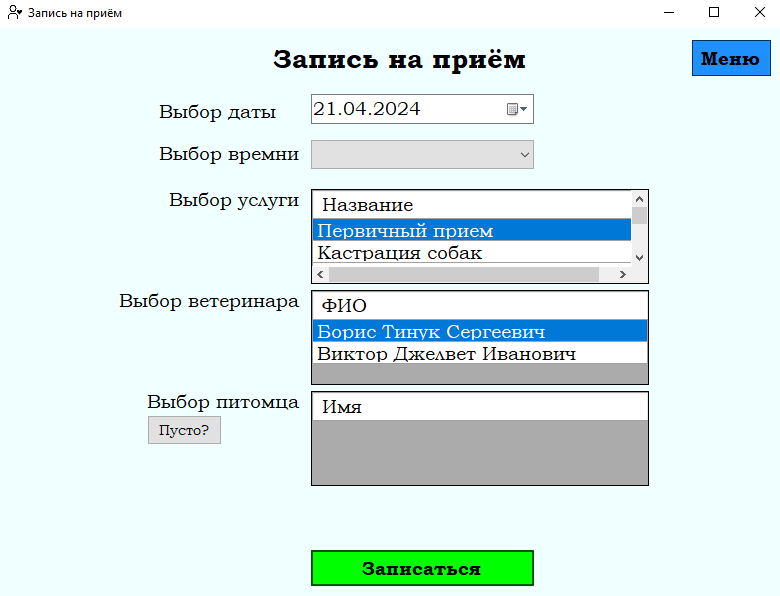


Рисунок 44 – Форма FormUserRecords

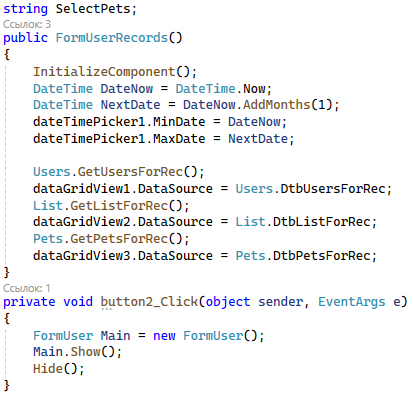


Рисунок 45 – Первая часть кода формы FormUserRecords

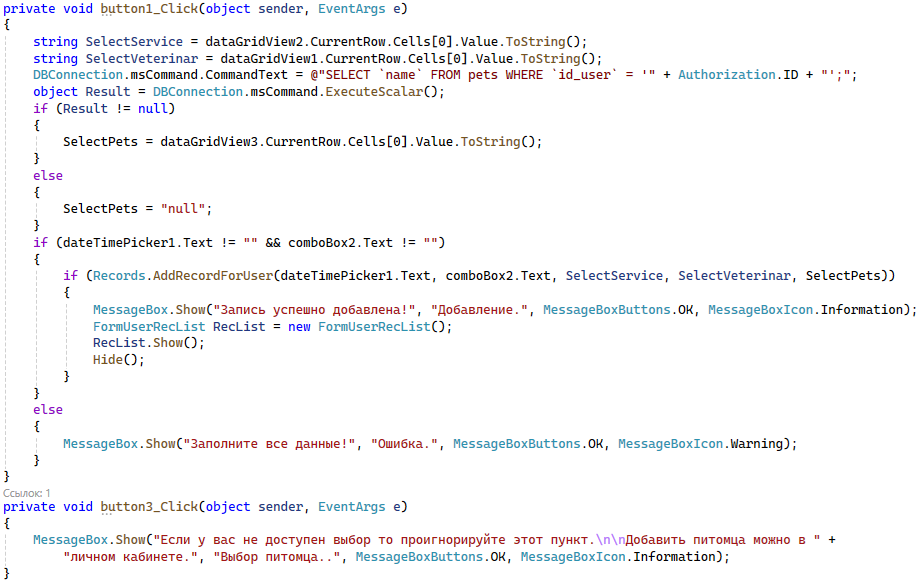


Рисунок 46 – Вторая часть кода формы FormUserRecords

5.2 Разработка базы данных ПП

В качестве базы данных для реализации программного продукта была выбрана база данных MySQL Workbench. Всего в базе данных располагается 6 таблиц, которые представлены на ER-модели, которую можно увидеть на рисунке 8.

Все таблицы спроектированы так, чтобы избежать избыточности, при этом достичь максимальной скорости работы, база данных нормализирована, и приведена к третьей нормальной форме, таблицы в которой связываются друг с другом посредством внешних ключей.

5.3 Разработка программы

Для связи базы данных с приложением потребуется создание подключения к базе данных, которая находится в MySQL Workbench. Для этого понадобилась библиотека MySql.Data, файл предоставлен на рисунке 47. А также нужно создать класс в C#, его код представлен на рисунке 48.



Рисунок 47 – Файл MySql.Data.dll



Рисунок 48 – Класс DBConnection

6. Документирование программного продукта

6.1 Руководство пользователя ПП

Функционал пользователя:

Первым открывается окно авторизации, оно представлено на рисунке 49. На нём у вас есть возможность зайти в свой аккаунт, если он уже создан, а если его ещё не существуете вы можете перейти на окно регистрации нажав на надпись «Ещё не зарегистрированы?»

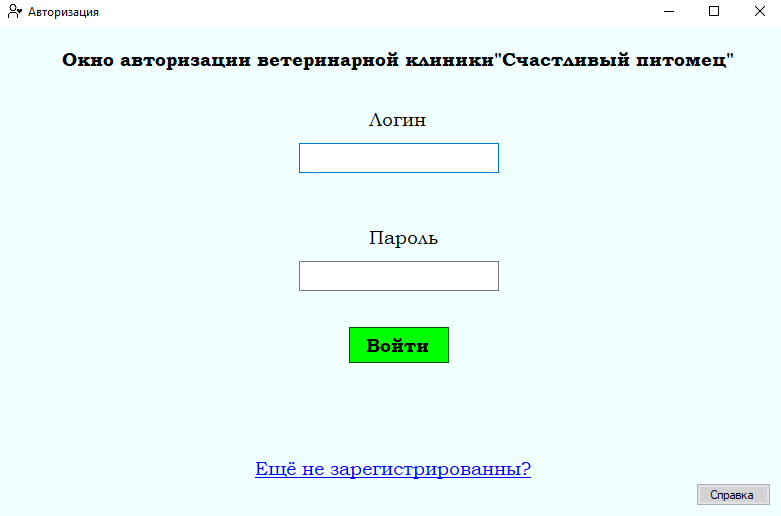


Рисунок 49 – Окно авторизации

Перейдя в окно регистрации, вы можете создать аккаунт. Для этого вам понадобиться заполнить все данные в правильном формате, программа уведомит вас в том случае если какие-либо данные будут не указанные, или указаны неверно. После регистрации вам вернёт в окно авторизации, где вы сможете войти в созданные аккаунт, ну или же вы можете нажать на надпись «Уже зарегистрированы? Авторизация», окно представлено на рисунке 50.

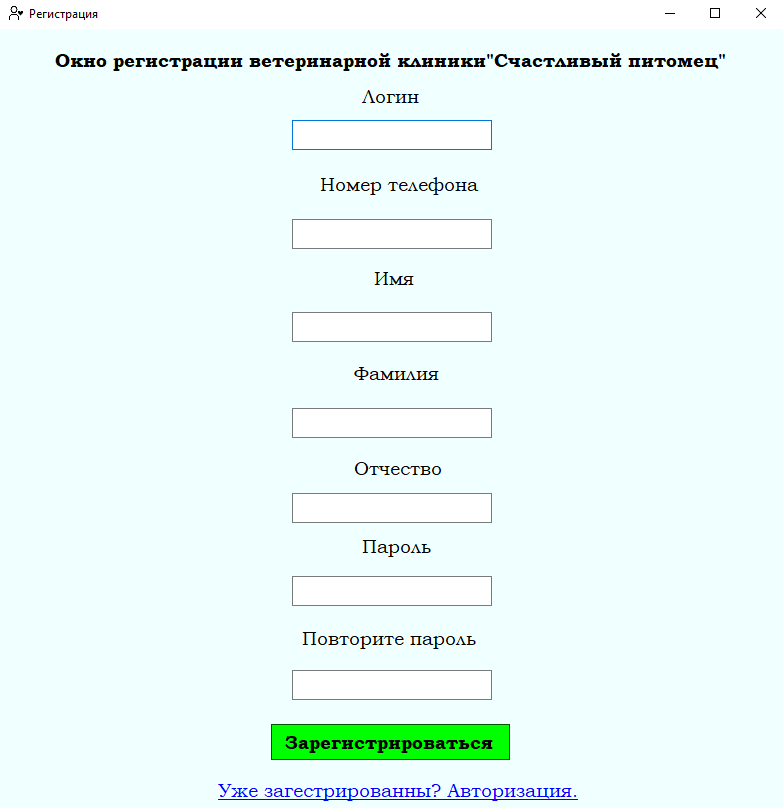


Рисунок 50 – Окно регистрации

В окне авторизации в нижнем правом углу есть кнопка «справка» которая пересылает нас в окно со справочной информацией.

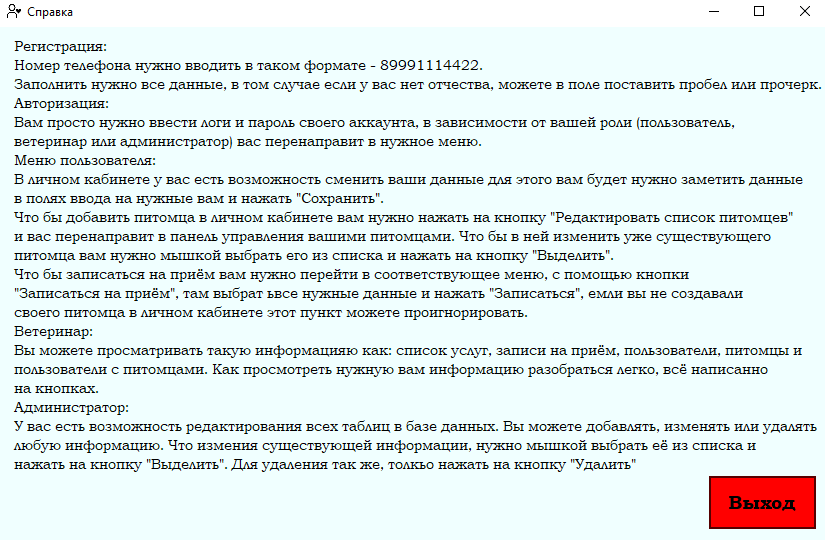


Рисунок 51 – Окно с справочной информацией

После входа в аккаунт ив зависимости от вашей роли, пользователь, ветеринар или администратор вы попадаете в нужную панель. Сейчас будет рассмотренно главное меню для обычно пользователя.

В главном меню вы можете увидеть ваш логин, он отображается в нижнем левом углу, сверху экрана будет большая надпись «Меню», под ней расположена актуальная информация по номеру телефона клиники, адрес, а также график работы. Функциональная часть представлена в правой части экрана, вы можете увидеть кнопки, которые позволяют перейти в окно с просмотром услуг, предоставляемых клиникой. Так же кнопку, отправляющую вас в личный кабинет. Ну и, конечно есть кнопка, которая перенаправляет вас в окно, где вы можете записаться на приём. Так же в самом низу списка есть кнопка позволяющая выйти из аккаунта. Окно представлено на рисунке 52.

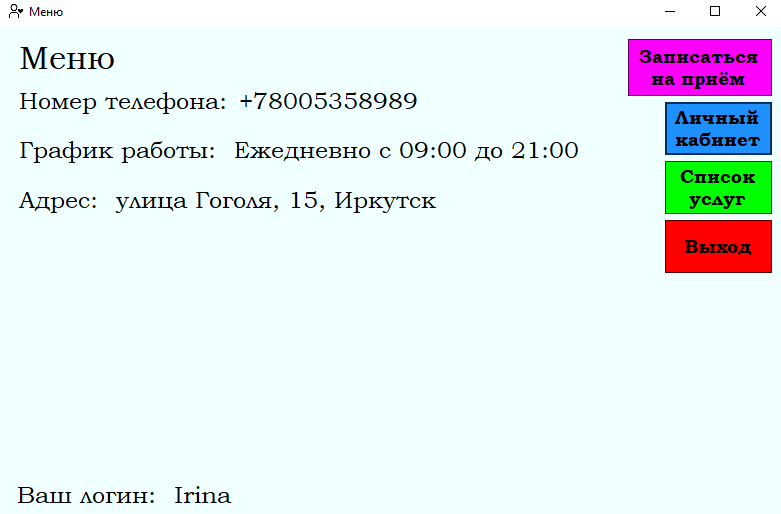


Рисунок 52 – Главное меню

Окно «Список услуг» содержит таблицу с информацией о всех услугах, а именно: название, цена и описание. Так же есть две кнопки которые предоставляют вернутся в главное меню или сразу перейти к записи на приём.

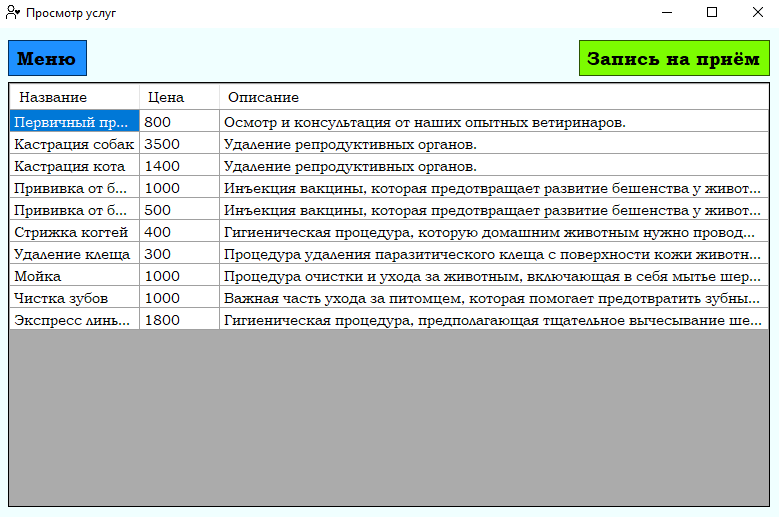


Рисунок 53 – Окно со списком услуг

Окно «Запись на приём» позволяет записаться на приём, для этого вам нужно указать данные, а именно: дата, время, название услуги, имя ветеринара и имя питомца. Выбор времени реализован с помощью удобного инструмента, у которого стоит ограничение на дату записи, а точнее запись открыта только на месяц вперёд. Выбор времени реализован через удобный список, в котором стоит ограничение на выбор времени раз в пятнадцать минут, а также ограничение на выбор во время обеда и ужина. Выбор услуги, ветеринара и питомца реализован одинаковым образом, это тоже выбор из списка. Необязательным пунктом является выбор питомца, но, если желание добавить своего питомца в базу данных у клиента есть, он может нажать на кнопку «Пусто?» где ему расскажут, где это сделать.

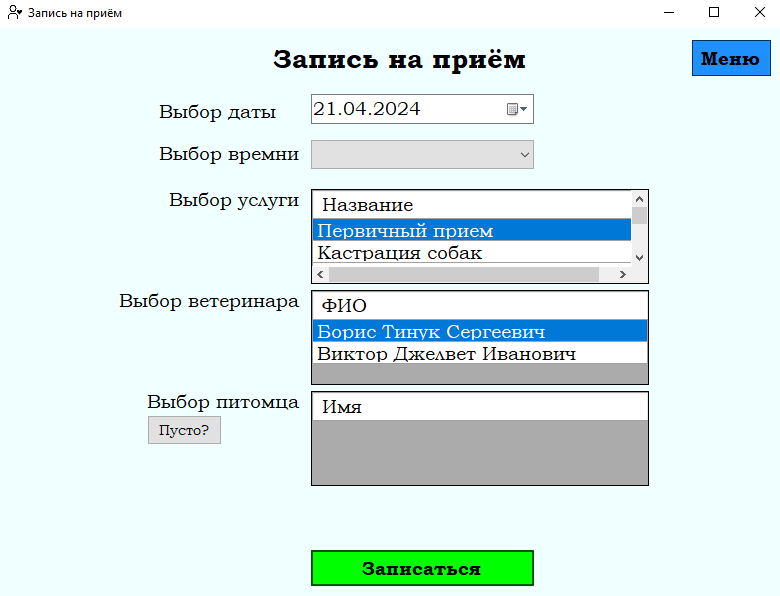


Рисунок 54 – Окно записи на приём

В личном кабинете вы можете, как и увидеть свои данные, так и поменять их. Для этого вам нужно правильно заполнить все поля данных, в случае ошибки программа вас предупредит, и нажать на кнопку «Сохранить». В верхнем правом углу расположены 2 кнопки: «Меню» и «Записи на приём», первая возвращает нас в меню, а вторая отсылает нас в окно просмотра ваших записей. По кнопке «Редактировать список питомцев» вы попадаете в окно с редактированием своих питомцев.

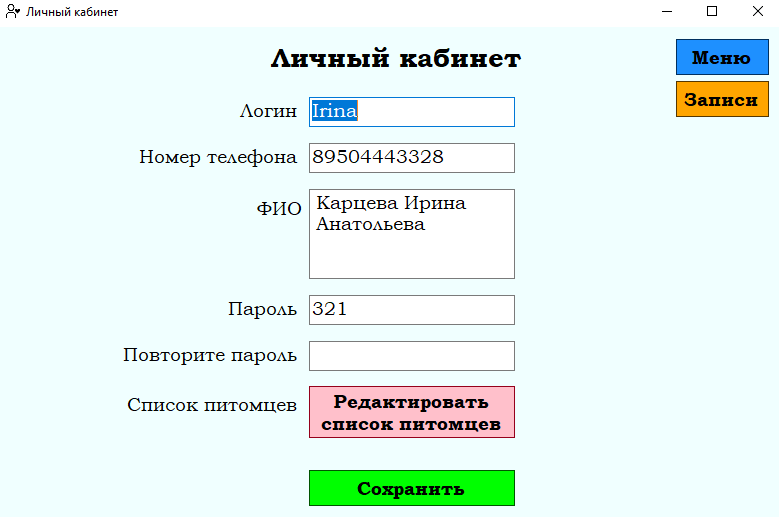


Рисунок 55 – Личный кабинет

Окно с редактированием своих питомцев содержит таблицу в нижней части экрана таблицу, в которой отображаются все ваши питомцы. Имеются два окна: «Добавление» и «Изменение», в окне добавления нужно указать имя, дату рождения и список болезней если они есть, в противном случае оставьте окно пустым, и нажмите «Добавить». Для изменения нужно выбрать мышкой питомца из таблицы в нижней части и нажать выделить, после в разделе «Изменения» заполнятся данные выбранного питомца, вы изменяете их и нажимаете кнопку «Изменить» и данные в таблице обновятся. Что бы удалить питомца выберите его в таблице из нижней части экрана мышкой и нажмите удалить.

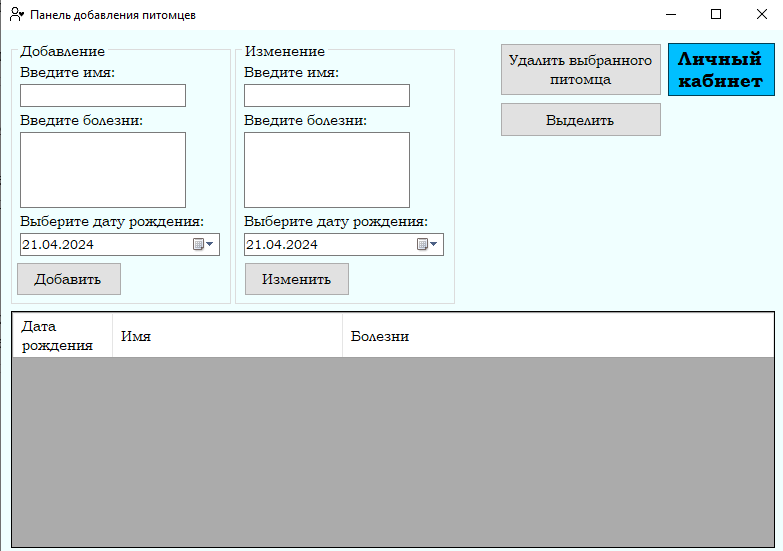


Рисунок 56 – Добавление питомца

Окно с просмотром ваших записей на приём позволяет их удалять, окно содержит 2 таблицы: одна для записей с указанным питомцам, вторая для записей без указанного питомца.

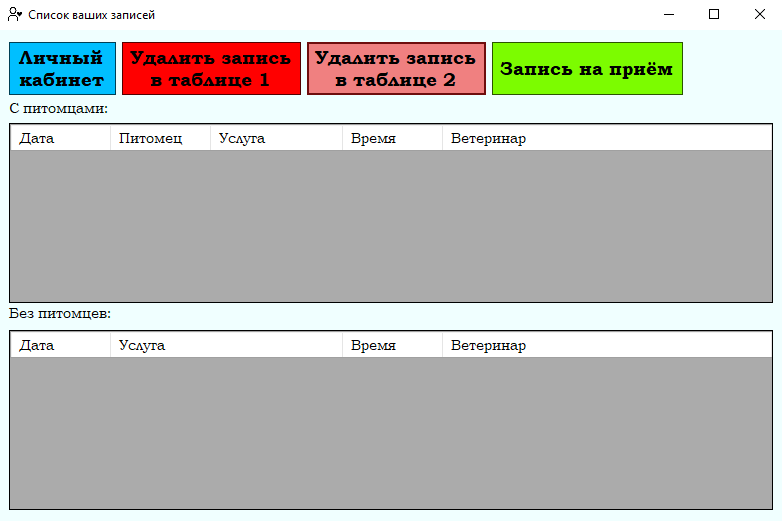


Рисунок 57 – Список записей пользователя

Функционал ветеринара:

В данном меню вы можете просматривать различные таблицы, функционал прост и понятен, из главного меню вы можете перейти в окно с просмотром услуг, где на весь экран таблица с услугами и рядом кнопка выхода обратно в меню, просмотр записей на приём работает так же. Для просмотра пользователей и питомцев есть отдельное окно, в котором что бы смотреть различные таблицы нужно нажимать на соответствующие этим таблица кнопки.

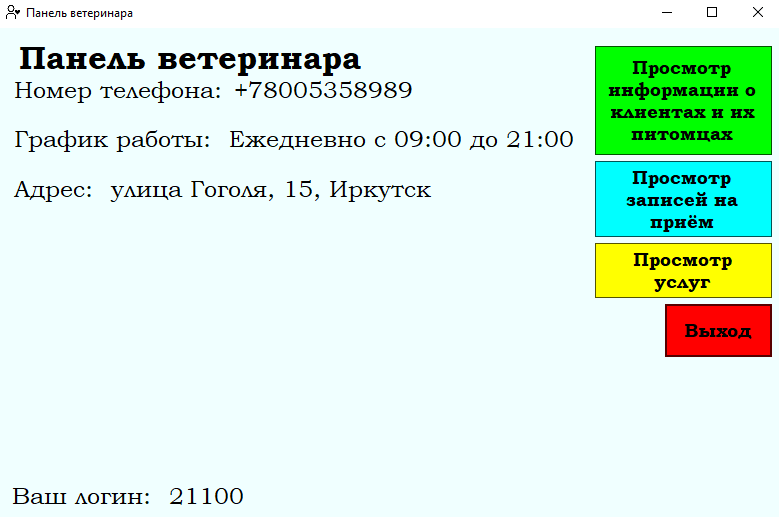


Рисунок 58 – Главное меню ветеринара

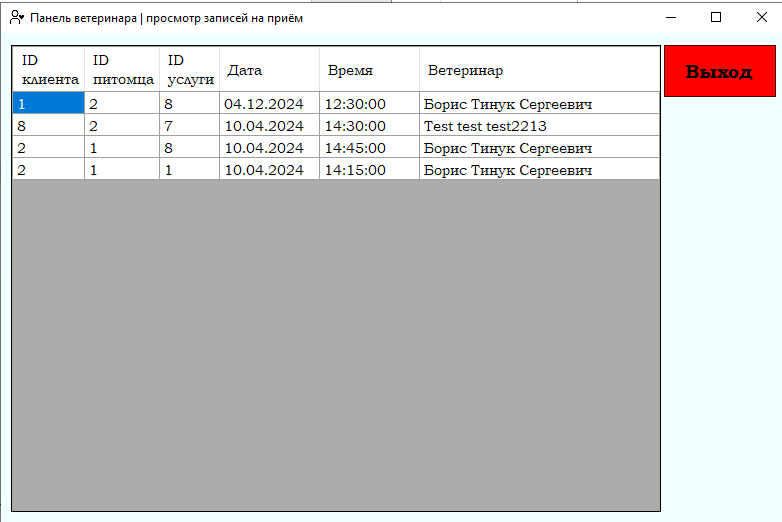


Рисунок 59 – Окно с просмотром записей на приём

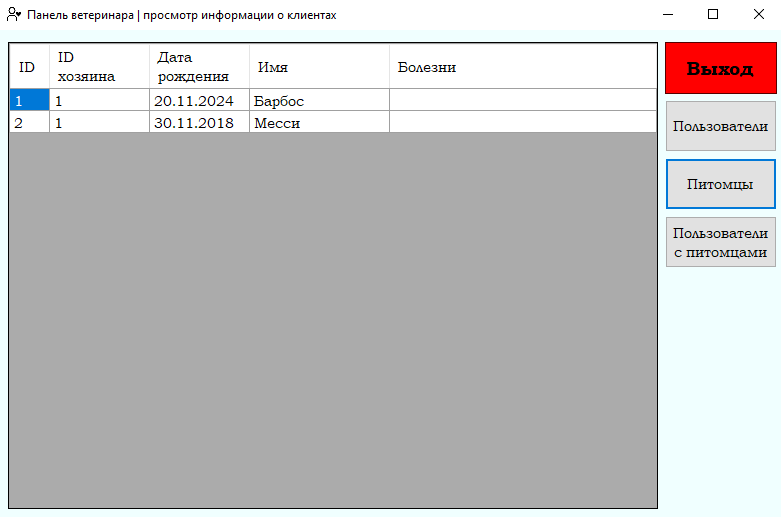


Рисунок 60 – Окно просмотра пользователей и питомцев

Функционал администратора:

Все окна администратора работают аналогично, поэтому разберём только одно с питомцами. Окно с редактированием питомцев содержит таблицу в нижней части экрана таблицу, в которой отображаются все питомцы. Имеются два окна: «Добавление» и «Изменение», в окне добавления нужно указать id пользователя, которому принадлежит питомец, имя, дату рождения и список болезней если они есть, в противном случае оставьте окно пустым, и нажмите «Добавить». Для изменения нужно выбрать мышкой питомца из таблицы в нижней части и нажать выделить, после в разделе «Изменения» заполнятся данные выбранного питомца, вы изменяете их и нажимаете кнопку «Изменить» и данные в таблице обновятся. Что бы удалить питомца выберите его в таблице из нижней части экрана мышкой и нажмите удалить.

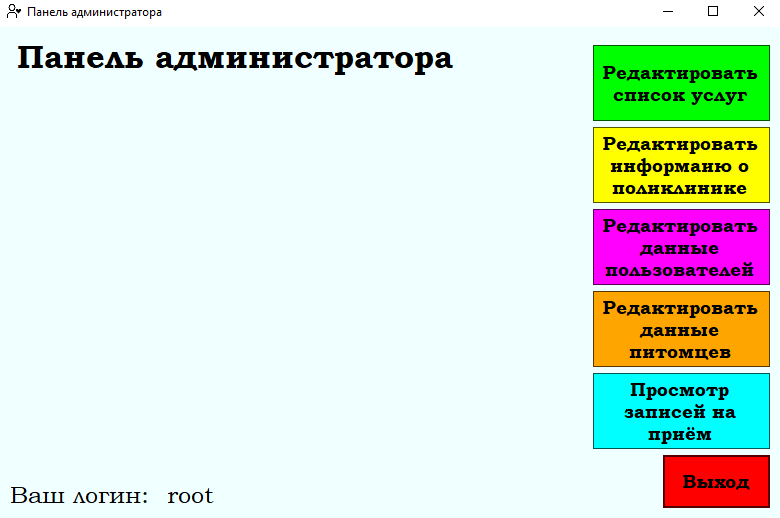


Рисунок 61 – Главное меню Администратора

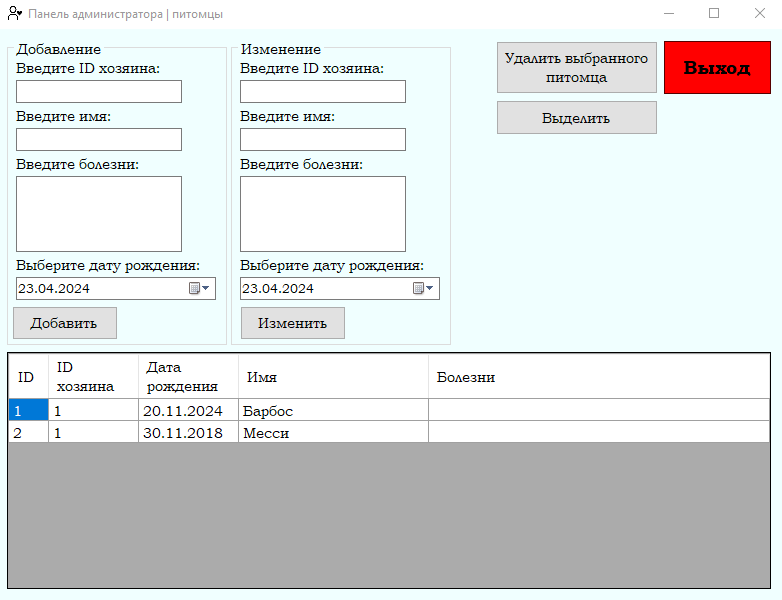


Рисунок 62 – Меню изменения таблицы питомцев у администратора.

Заключение

В ходе выполнения курсовой работы было создано приложение «Ветеринарная клиника», которое позволяет упростить процесс взаимодействия пользователя, ветеринара и администратора. Для приложения был определен и реализован следующий функционал:

* Регистрация;
* Авторизация по ролям;
* Просмотр такой информации о клинике как: номер телефона, адрес и график работы;
* Изменение данных в личном кабинете, для пользователя;
* Запись на приём с выбором даты, времени, услуги, ветеринара и питомца, для пользователя;
* Просмотр списка своих записей на приём, для пользователя;
* Добавление питомцев с возможностью выбора даты рождения, имени и списка болезней, для пользователя;
* Просмотр списка услуг и информации о них для пользователя и ветеринара;
* Просмотр такой информации как ID, ФИО и номер телефона о всех пользователях приложения, для ветеринара;
* Просмотр информации о питомцах, для ветеринара;
* Просмотр списка всех записей на приём, для ветеринара;
* Просмотр и редактирование списка услуг, для администратора;
* Просмотр и редактирование информации о клинике, для администратора;
* Просмотр и редактирование списка пользователей, для администратора;
* Просмотр и редактирование списка питомцев, для администратора;
* Просмотр и редактирование списка записей на приём, для администратора.

В процессе анализа инструментальных средств разработки были рассмотрены множество вариантов реализации приложения, но выбран был язык программирования C#.

Программный продукт обладает большим функционалам, а также имеет интуитивный интерфейс.

Главная сложность в процессе разработки программного продукта это настройка корректной работы приложения с базой данных. Данная проблема была решена путём изучения дополнительных материалов.

Программный продукт может расширятся путём добавления нового функционала и совершенствование интерфейса.

Все поставленные цели и задачи курсовой работы были успешно выполнены.

Список используемых материалов

1 Csharp – Знакомство с Visual Studio – URL: https:// https://csharp.webdelphi.ru/znakomstvo-s-visual-studio/ (Дата обращения 11.02.24). – Текст: электронный.

2 Computing at School – Text Adventure Game using C# and Winforms – URL: https://www.computingatschollt.org/ru-ru/dotnet/csharp/fundamentals/program-structure (Дата обращения 12.03.24). – Текст: электронный.

3 Stack Overflow – Fill Datagridview with MySQL data – URL: https://stackoverflow.com/questions/12904854/fill-datagridview-with-mysql-data (Дата обращения 15.03.24). – Текст: электронный

4 Guru99 – C# Windows Учебное пособие по применению форм с примером – URL: https://www.guru99.com/ru/c-sharp-windows-forms-application.html (Дата обращения 30.01.24). – Текст: электронный.

5 CyberForum.ru – MySQL. Вывести в dataGridView таблицу – URL: https://www.cyberforum.ru/ado-net/thread711746.html (Дата обращения 15.03.24). – Текст: электронный.

6 Microsoft – Использование .NET (C#) для подключения в База данных для MySQL – URL: https://learn.microsoft.com/ru-ru/azure/mysql/single-server/connect-csharp (Дата обращения 22.02.24). – Текст: электронный.

7 Microsoft – Общая структура программы C# – URL: https://learn.microsoft.com/ru-ru/dotnet/csharp/fundamentals/program-structure (Дата обращения 27.01.24). – Текст: электронный.

8 StudFies – Windows Forms – URL: https://studfile.net/preview/842423/ (Дата обращения 03.03.24). – Текст: электронный.

9 SQLite или MySQL – SQLite или MySQL – В чем различие – URL: https://ilso.ru/sqlite-vs-mysql/ (Дата обращения 03.02.24). – Текст: электронный.

10 Интернет технологии.ру – Команды MySQL – URL: https://www.internet-technologies.ru/articles/komandy-mysql.html (Дата обращения 16.02.24). – Текст: электронный.

Приложение А – Техническое задание

**Министерство образования Иркутской области**

Государственное бюджетное профессиональное образовательное учреждение

Иркутской области

«Иркутский авиационный техникум»

(ГБПОУИО «ИАТ»)

|  |  |
| --- | --- |
|  |  |

ТЕХНИЧЕСКОЕ ЗАДАНИЕ

ПРИЛОЖЕНИЯ

«ВЕТЕРИНАРНАЯ КЛИНИКА»

|  |  |  |
| --- | --- | --- |
| Руководитель: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (Н.Р. Карпова) |
| Студент: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (Л.С. Чиин) |

Иркутск 2024

**1 Общие сведения**

Наименование работы: приложение «Ветеринарная клиника»

Исполнитель: студент Иркутского авиационного техникума, отделен, группы ИС-21-1, Чиин Л.С.

Разработка приложения проходит в рамках курсовой работы.

Сроки разработки программного продукта с 29.01.2024 по 10.04.2024 года.

**2 Цели и назначение создания автоматизированной системы**

Целью курсовой работы является создание приложения «Ветеринарная клиника», которое включает в себя возможности просмотра информации о клинике, о её ветеринарах и предоставляемых услугах, так же запись на приём.

В данном приложении реализуется следующий функционал:

1. Системный администратор:

* Имеет доступ к базе данных и ей содержимому.
* Добавляет услуги, изменяет текущую информацию на актуальную.
* Составляет или редактирует расписания работы специалистов.

1. Клиент:

* Начинает взаимодействие с приложением, проходит авторизацию или регистрацию.
* Просматривает информацию о клинике и её услугах.
* Начинает запись на приём, выбирает удобное время из доступного, выбирает ветеринара и подтверждает запись.

1. Ветеринар:

* Просмотр данных о клиентах и список приёмов.

**3 Характеристика объекта автоматизации**

Данный программный продукт разрабатывается для облегчения ведения бизнеса ветеринарной клиники. При помощи продукта будет проще будет записаться на приём и узнать нужную информации о клинике.

4 Требования к системе в целом

## **4.1 Требования к структуре и функционированию приложения**

Функции приложения:

1. Меню «Авторизации»:
   1. авторизация пользователей.
2. Меню «Регистрация»:
   1. регистрация пользователей.
3. Меню «Главная» для пользователей:
   1. Информация о клинике;
   2. Информация о ветеринарах;
   3. Информация о услугах;
   4. Начало записи на приём;
4. Меню «Запись на приём»:
   1. Выбор ветеринара;
   2. Выбор времени.
   3. Указание питомца;
5. Меню «Главная» для ветеринара:
   1. Просмотр списка приёмов;
   2. Просмотр информации о клиентах из базы данных клиники;
6. Меню «Главная» для администратора:
   1. Просмотр списка приёмов, возможность редактирования.
   2. Просмотр списка услуг, возможность редактирования.
   3. Просмотр списка ветеринаров, возможность редактирования.
   4. Просмотр информации о клинике, возможность редактирования.

**4.2 Требования к надежности**

Для обеспечения надежности необходимо проверять корректность получаемых данных и реализовать валидность полей. Входные данные поступают в виде значений c клавиатуры. Эти значения отображаются в отдельных полях ввода.

**4.3 Требования к безопасности**

Для обеспечения безопасности, необходимо реализовать разграничение прав доступа.

**4.4 Требования к эксплуатации, техническому обслуживанию, ремонту и** **хранению компонентов системы**

К серверной части приложения:

1. процессор: Intel Pentium 4 или более поздней версии с поддержкой SSE2;
2. оперативная память: 1 Гб;
3. 5 Гб на HDD;
4. клавиатура, мышь;
5. встроенное видео ядро;
6. стабильный доступ в интернет;
7. наличие базы данных SQL Workbeanch;
8. операционная система: Windows 7/8/10.

К клиентской части приложения:

1. процессор 1 ядро 1.3 ГГц и более сильные;
2. оперативная память: 1 ГБ;
3. 1 Гб на SSD или HDD;
4. встроенное видео ядро;
5. стабильный доступ в интернет;
6. Операционная система: Windows 7/8/10.

**5 Требования к документированию**

Основным документам, регламентирующими использование приложения является руководство пользователя.

Основным документам, регламентирующими разработку приложения является техническое задание.

**6 Состав и содержание работ по созданию системы**

В таблице 1 представлены плановые сроки начала и окончания работы по созданию приложения.

Таблица 1 – Этапы выполнения разработки.

|  |  |  |
| --- | --- | --- |
| Наименование этапов | Срок | Результат |
| Предпроектное исследование предметной области | до 29.01.2024 | Описана предметная область |
| Разработка технического задания (выбор архитектуры программного обеспечения, выбор типа пользовательского интерфейса, выбор языка и среды программирования) | до 05.02.2024 | ТЗ |
| Проектирование программного обеспечения.  (разработка структурной и функциональной схемы ПО, проектирование базы данных (инфологическое, ER-модель, физическая модель) | до 26.03.2024 | Проектирование (моделей) |
| Разработка (программирование) и отладка программного продукта | до 10.04.2024 | Программный код |
| Составление программной документации (оформление руководства пользователя) | До 15.04.2024 | Руководство пользователь |

Приложение Б – Листинг FormLogin.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Veterinary

{

public partial class FormLogin : Form

{

public FormLogin()

{

InitializeComponent();

DBConnection.ConnectionDB();

}

private void button1\_Click(object sender, EventArgs e)

{

if (textBox1.Text != "" && textBox2.Text != "")

{

Authorization.Authorizations(textBox1.Text, textBox2.Text);

switch (Authorization.Role)

{

case null:

{

MessageBox.Show("Такого аккаунта не существует или пароль введён неверно!",

"Проверьте данные и начните снова!");

break;

}

case "Администратор":

{

Authorization.User = textBox1.Text;

MessageBox.Show(textBox1.Text + ", добро пожаловать в меню администратора!",

"Успешный вход в панель администратора.",

MessageBoxButtons.OK, MessageBoxIcon.Information);

this.Hide();

FormAdm fA = new FormAdm();

fA.Show();

break;

}

case "Ветеринар":

{

Authorization.User = textBox1.Text;

MessageBox.Show(textBox1.Text + ", добро пожаловать в меню ветеринара!",

"Успешный вход в панель ветеринара.",

MessageBoxButtons.OK, MessageBoxIcon.Information);

this.Hide();

FormVet fV = new FormVet();

fV.Show();

break;

}

case "Пользователь":

{

Authorization.User = textBox1.Text;

MessageBox.Show(textBox1.Text + ", добро пожаловать в меню пользователя!",

"Успешный вход в меню.",

MessageBoxButtons.OK, MessageBoxIcon.Information);

this.Hide();

FormUser f4 = new FormUser();

f4.Show();

break;

}

}

}

else

{

MessageBox.Show("Заполните поля с даннными!");

}

}

private void linkLabel1\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

FormRegister f = new FormRegister();

f.Show();

this.Hide();

}

private void textBox2\_KeyPress(object sender, KeyPressEventArgs e)

{

if (e.KeyChar == '\r')

{

button1\_Click(sender, e);

}

}

private void button1\_Click\_1(object sender, EventArgs e)

{

FormHelp FH = new FormHelp();

FH.Show();

Hide();

}

}

}