МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет	*Кафедра*
вычислительной техники	МОиПЭВМ

Направление подготовки 09.03.02 «Информационные системы и технологии»

БАКАЛАВРСКАЯ РАБОТА

на тему

Информационная система клуба федерации ездового спорта

Студент _____ Казикина Дарья Александровна
(подпись, дата) (ФИО полностью)

Руководитель _____ Кольчугина Е.А.
(подпись, дата) (фамилия, инициалы)

Нормоконтролер _____ Такташкин Д.В.
(подпись, дата) (фамилия, инициалы)

Работа допущена к защите (протокол заседания кафедры от _____№_____)

Заведующий кафедрой _____ Макарычев П.П.
(подпись) (фамилия, инициалы)

Работа защищена с отметкой _____ (протокол заседания ГЭК от _____№____)

Секретарь ГЭК _____ Попова Н.А.
(подпись) (фамилия, инициалы)

Пенза, 2021

Федеральное государственное бюджетное образовательное учреждение высшего образования
ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет	*Кафедра*
Вычислительной техники	МО и ПЭВМ

«Утверждаю»
Заведующий кафедрой
_____П.П. Макарычев
«__» _____2021 г

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
БАКАЛАВРА

1. Студент Казикина Дарья Александровна гр. 17ви1 факультета ВТ
<Фамилия, имя, отчество полностью>
2. Тема работы Информационная система клуба федерации ездового спорта

Тема утверждена приказом ПГУ №_____от "_____"_____2021 г.

2. Руководитель работы к.т.н., доцент кафедры «МОиПЭВМ» Кольчугина Е.А.
3. Задание на работу (назначение разработки, исходные данные и т.п.)

1) Назначение: программное обеспечение предназначено для регистрации членов и учета информации о соревновательной деятельности собак

2) Среда выполнения: операционная система Windows, начиная с версии 7.

Средства разработки: языки программирования C#, SQL; инструментальные средства Microsoft Visual Studio, Microsoft SQL Server.

Технологии разработки: ООП.

3) Основные функции автоматизированной информационной системы:

а) регистрация новых членов клуба

б) ведение статистики по оплате членских взносов;

в) учет собак входящих в клуб ;

г) формирование медицинских паспортов собак;

д) фиксирование наград каждой собаки;

е) редактирование сведений о собаках;

ж) редактирование сведений о членах клуба;

з) управление соревнованиями проводимыми федерацией ездового сорта;

и) учет участия собак клуба в соревнованиях;

к) учет заездов на соревнованиях по ездовому спорту;

л) *формирование аналитического отчета об денежных средствах полученных их членских взносов*

4. Перечень подлежащих разработке вопросов

Анализ предметной области

Анализ требований

Проектирование

Реализация

Контроль качества ПО:
- *Тестирование*
- *Анализ качества кода*

План разработки и оценка бюджета

5. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	Анализ предметной области и требований	10	07.05.2021 – 09.05. 2021	
2	Проектирование	25	10.05. 2021 – 15.05.2021	
3	Реализация	35	16.05.2021 – 29.05.2021	
4	Контроль качества	15	30.05.2021 – 02.06.2021	
5	План разработки бюджета	5	03.06.2021 – 05.06.2021	
6	Оформление пояснительной записки	10	06.06.2021 – 10.06.2021	

Дата выдачи задания «___» _____2021 г.

Руководитель бакалаврской работы _____
(подпись, дата)

Задание к исполнению принял студент _____
(подпись, дата)

Работу к защите допустить

Декан факультета _____
(подпись, дата)

Реферат

Пояснительная записка содержит 113 листов, 44 рисунка, 16 использованных источников.

АВТОМАТИЗИРОВАННАЯ ИНФОРМАЦИОННАЯ СИСТЕМА, БАЗА ДАННЫХ, ПРЕДСЕДАТЕЛЬ КЛУБА, КИНОЛОГ, СЕРВЕР, КЛИЕНТ

Объект разработки – база данных, которая содержит данные об определенной предметной области (данные о членах и собаках клуба ездового спорта) и клиентское приложение для управления этими данными в базе данных.

Цель работы – разработать клиентское приложения для сервера базы данных с возможностью изменять, удалять, добавлять новые объекты определенной предметной области в базу данных.

Технология разработки – для создания серверной части используется Microsoft SQL Server, для создания клиентской части используется среда программирования MS Visual Studio, Windows Form, C#, SQL.

Результаты работы – в ходе выполнения бакалаврской работы были созданы серверная и клиентская часть приложений. Работа между ними налажена. Все требования к разработке приложения соблюдены.

Содержание

Введение 6

1 Анализ предметной области и требований 8

1.1 Анализ предметной области 8

1.2 Анализ программных аналогов 9

1.3 Постановка задачи 14

1.4 Анализ бизнес-процессов предметной области 15

1.5 Анализ требований 26

1.5.1 Анализ функциональных требований 26

1.5.2 Анализ нефункциональных требований 32

1.6 Выбор инструментальных средств 33

1.7 Системные программные средства 33

2 Проектирование информационной системы клуба федерации ездового спорта 35

2.1 Базовая архитектура системы 35

2.2 Логическая модель базы данных 35

2.3 Проектирование пользовательского интерфейса 38

2.4 Проектирование форм отчетов 43

2.5 Технология разработки 46

2.5.1 Шаблон проектирования «Адаптер» 46

2.6 Проектирование программных средств 47

2.6.1 Общее проектирование системы 47

3 Реализация информационной системы клуба федерации ездового спорта 50

3.1 Физическая модель базы данных 50

3.2 Реализация программных средств 51

3.2.1 Реализация подсистемы «Председатель» 52

3.2.2 Представление развертывания 54

4 Тестирование информационной системы клуба федерации ездового спорта 56

4.1.1 Функциональное тестирование 56

4.1.2 Модульное тестирование 59

4.2 Метрики кода 60

5 Планирование проекта и оценка бюджета 64

Заключение 72

Список использованных источников 73

Приложение А. Глоссарий 75

Приложение Б. Листинг программы 77

Приложение В. Листинг классов базы данных 93

Приложение Г. Результаты тестирования 99

Приложение Д. Листинг тестового проекта 103

Введение

Клуб федерации ездового спорта – общественная организация, ответственная за проведение соревнований по различным дисциплинам ездового спорта. Также клуб – кинологическая организация, которая располагает систематизированной информацией о ездовых собаках, и использует ее с самыми, различными целями в частности для отслеживание медицинских паспортов собак и управлениями соревнованиями проводимыми клубами

Деятельность клуба ездового спорта формируется из нескольких компонентов основных компонентов: работа по учету членов клуба и их членских взносов книг, учет данных о ездовых собаках, управление соревновательной деятельностью.

Главным преимуществом автоматизирования данной сферы является предоставление членам клуба полных и достоверных сведений о состоянии денежных взносов, а также возможности быстрой работе по управлению информацией

Актуальность данной темы проявляется в использовании современных подходов для систематизации и использования информации. Из-за повсеместного внедрения информационных технологий, есть возможность сделать работу общественной организации клуба ездового спорта собак более эффективным благодаря внесению оперативной информации в базу данных, для увеличения возможностей по-быстрому и точному отбору информации, которая используется в целях систематизации и управления

Целесообразность реализации выпускной квалифицированной работы обусловлена отсутствием автоматизированных информационных систем (АИС), по данной узкоспециализированной отрасли и удовлетворяющих потребности конечного пользователя – администрации общественной организации клуба федерации ездового спорта.

Цель данной выпускной квалификационной работы – разработка ИС, которая позволит многократно увеличить эффективность выполнения происходящих в клубе ездового спорта процессов.

1 Анализ предметной области и требований

1.1 Анализ предметной области

Предметной областью разработки является информационная система клуба федерации ездового спорта, представляющая собой совокупность данных о членах клуба, собаках членов клуба и соревнованиях, проводимых клубом в рамках календаря соревнований по ездовому спорту библиотек, находящихся на территории города. Членами клуба являются любители собаководства, оплачивающие членские взносы за определенные периоды членства в клубе. Члены клуба могут регистрировать в клубе своих ездовых собак, при этом клуб ведет учет информации по медицинским паспортам собак их наградам и титулам. Кроме того, клуб ведет селекционную работу, формирует различные ездовые упряжки для различных видов ездового спорта, что дает возможность анализа, для повышения результативности животных на соревнованиях.

Как правило, членство в клубе можно регистрировать по номерам членских книжек, а учет собак вести по номерам клеймения животных.

Кроме этого оперативно ведутся изменения в медицинских паспортах собак, туда добавляются данные о прививках, перенесенных болезнях, готовности к участию в разведении.

Для дальнейшей селекционной работы крайне важен и учет наград и титулов у собак клуба, что позволяет выявлять и развивать лучшие породные качества у собак

Клуб также ведет активную соревновательную деятельность и поэтому для управления заездами в соревнованиях по различным видам ездового спорта всегда актуальна информация о результативности и медицинском состоянии собак.

Кроме того, существует большое число разновидностей ездового спорта как по земле, так и по снежному покрытиям, информации имеющаяся

у членов клуба позволяет оптимально распределять животных пол видам, типам, саней и по количеству собак в упряжках

1.2 Анализ программных аналогов

В настоящее время существует некоторое количество как зарубежных, так и отечественных аналогов, автоматизирующих деятельность клубов собаководства. Преимуществами данных систем является наличие большого количества функций и поддержка работы с базами данных. Основным недостатком является отсутствие специализации таких программ под клубную деятельность, связанную с ездовым спортом, практически все имеющиеся программы необходимо приспосабливаться и то не полной мере, к соревновательной деятельности ездового спорта. Кроме того, многие функции таких программ не находят себе вообще некого применения, а некоторые явно избыточны.

Наиболее распространенными являются следующие системы: «Кинолог_V.0n», «Собаководство », «Dogs-info», «Работа - заводчика», «Dog Training – American Kennel Club».

«Кинолог» на рынке с 2014 года, является свободно распространяемым программным обеспечением, для отслеживания результативности рабочих качеств охотничьих собак.

Комплекс «Кинолог» внедрен в большое количество клубов и федераций охотничьего собаководства как на территории нашей страны, так и странах СНГ

Можно отметить, что система имеет довольно сложный интерфейс, имеющий множество полей для заполнения, из которых реально заполняется лишь половина, да и то требуется высокая квалификации по сопровождению данного программного обеспечения. Пользователь может потратить много времени, выбирая поля, которые необходимо заполнить.

Преимуществами данной системы являются:

– большое число функций, реализованное посредством множества модулей (автоматизированных рабочих мест);

– защита от несанкционированного доступа к базам данных;

– поддержка копирования данных из одной базы данных в другую;

– также есть возможность миграции данных в наиболее популярные форматы.

«Кинолог» является доступным приложением и использует в качестве СУБД Microsoft SQL-Server.

Форма учета данных о породе русская гончая в системе «Кинолог» представлена на рисунке 1.

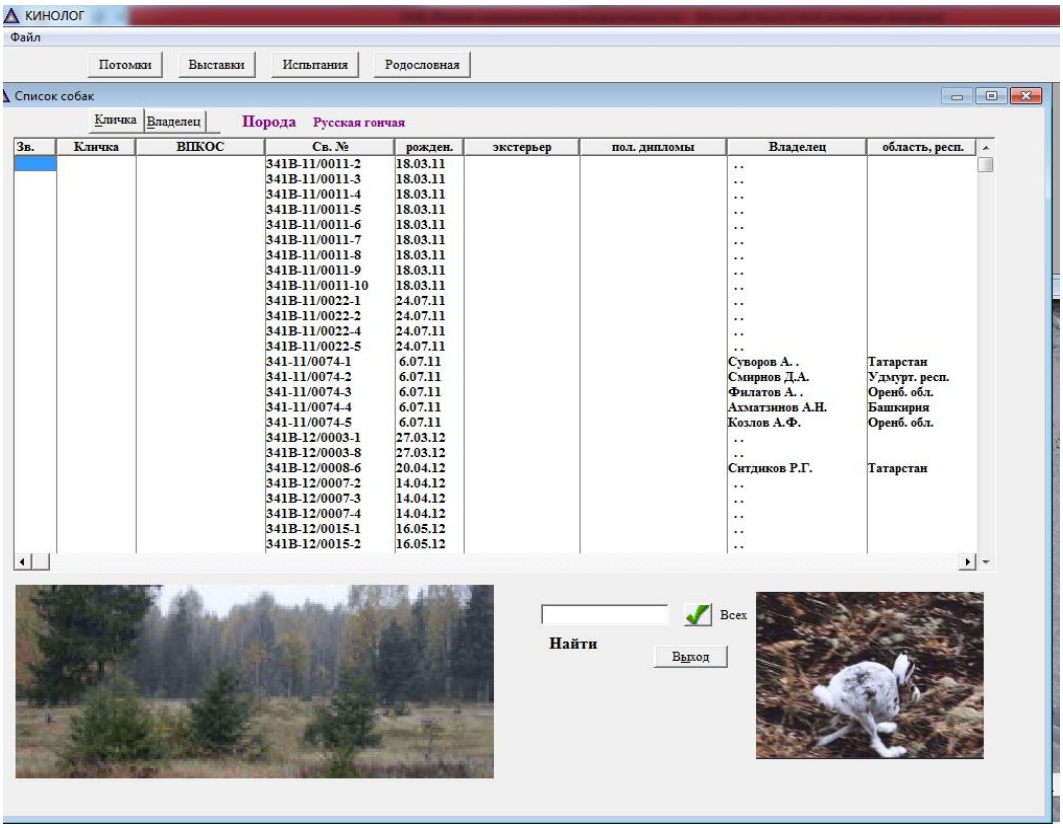


Рисунок 1 – Форма заполнения данных в системе «Кинолог»

Отчет по линиям родословных для породы «Русская гончая» в программе «Кинолог» представлена на рисунке 2.

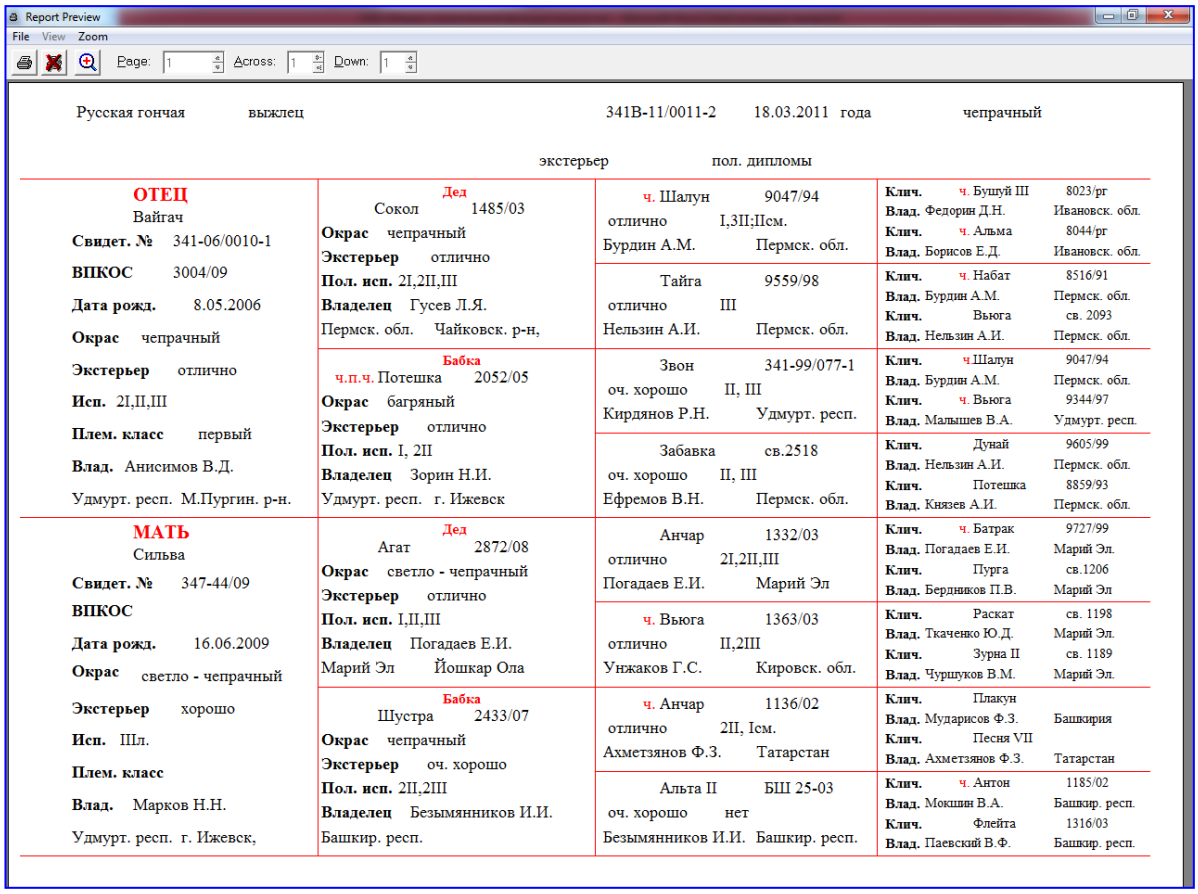


Рисунок 2 – Отчет по линиям родословных в программе «Кинолог» и заполнения данных в системе «Кинолог»

ИС «Собаководство» также относится к свободно распространяемому программному обеспечению, начало разработки 2011 год, последнее обновление 28.01.2020 г., то есть программное обеспечение, находится в актуальном состоянии

Среди преимуществ можно отметить структурированное хранение данных, использование технологии online справки, открытость значительной части кода, и открытый доступ к базам данных пород.

Система ориентирована на проведение экстерьерных выставок и ведения селекционной работы.

Форма заполнения данных ИС «Собаководство» представлена на рисунке 3.

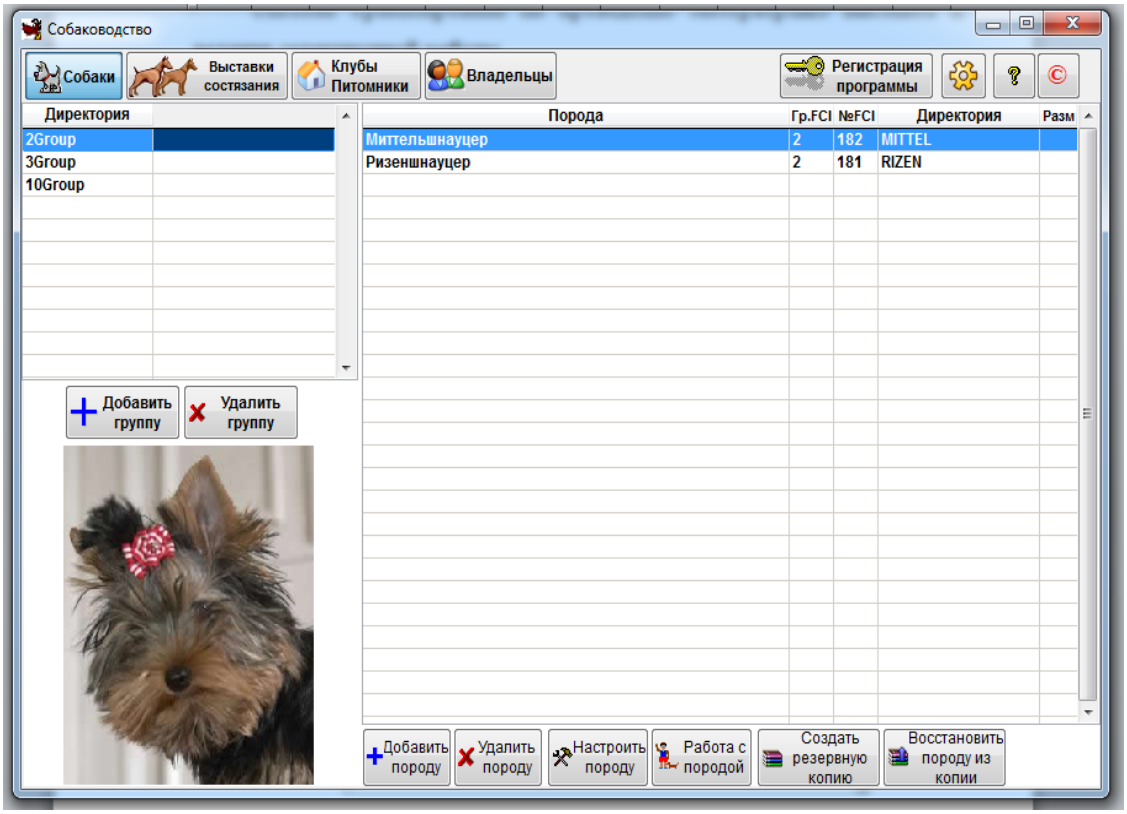


Рисунок 3 – Форма заполнения данных в ИС «Собаководство»

Система «Собаководство» обладают и простым и удобным для кинолога пользовательским интерфейсом, имеют огромный функционал, позволяет заносить различные данные на собак: кличку, владельца, клеймо, дату рождения, окрас, рабочие дипломы, и т.д. Программа предназначена для избавления кинологов от тяжелой рутинной работы и автоматического формирования всех кинологических документов, но нет функций учета достижений собак и нет специфических возможностей для ездового спорта.

Также система имеет платные подключаемые дополнительно модули.

Форма электронного каталога системы ИС «Собаководство» представлена на рисунке 4.

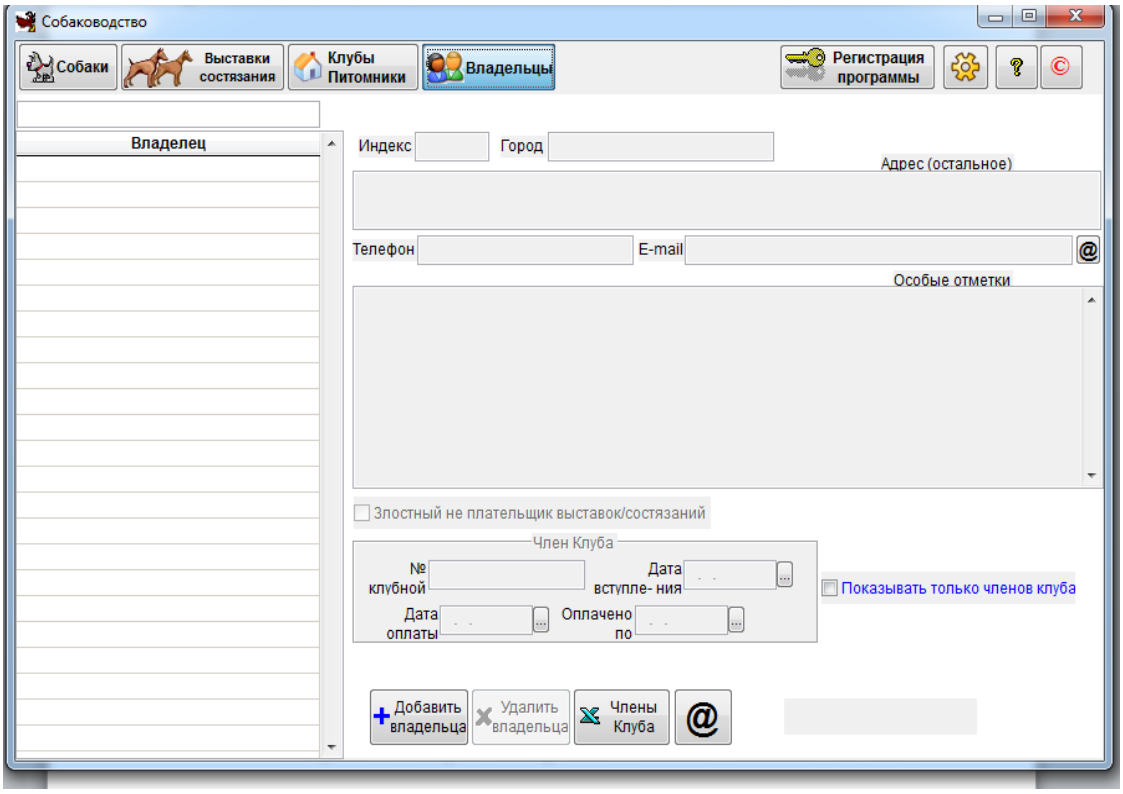


Рисунок 4 – Форма электронного учета системы ИС «Собаководство»

Всемирно известный ресурс Dog Training – American Kennel Club является огромной международной веб платформой, где большая часть добавочного функционала является платной или предоставляется по платной подписке, англоязычный интерфейс.

American Kennel Club – это реестр чистокровных пород собак в Соединенных Штатах. Помимо ведения своего родословного реестра, этот клуб собаководства также продвигает и санкционирует мероприятия для чистокровных собак, включая выставку собак вестминстерского клуба собаководства, ежегодное мероприятие, которое предшествует официальному формированию АКС, Национальной выставке собак и Национальному чемпионату АКС.

Форма регистрация собак в системе «Dog Training – American Kennel Club» представлена на рисунке 5 (перевод страницы с помощью браузера Хром).

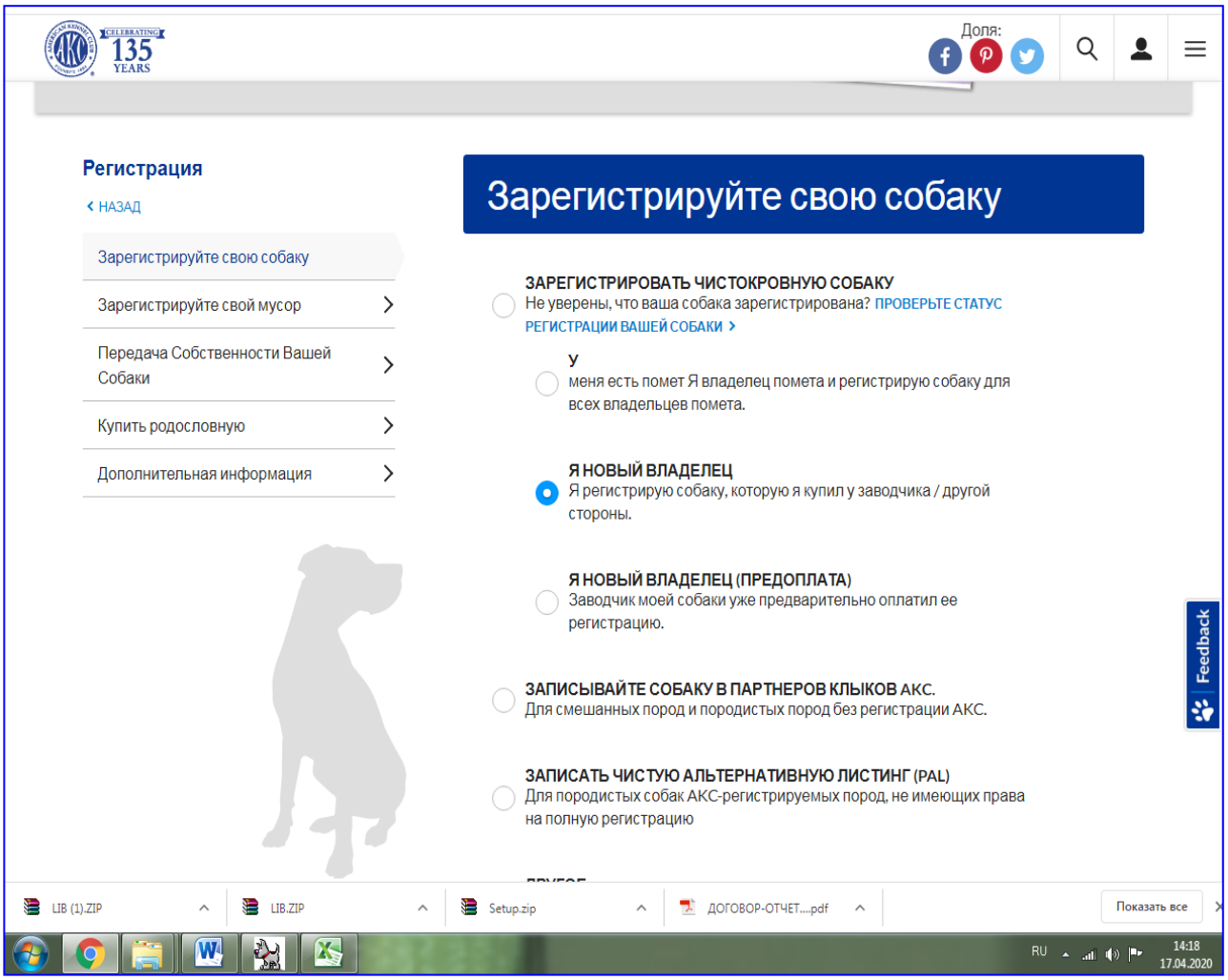


Рисунок 5 – Форма регистрации в «Dog Training – American Kennel Club»

В ходе анализа программных аналогов были сделаны выводы о достоинствах и недостатках, которые будут учтены при разработке собственной автоматизированной библиотечной информационной системы.

1.3 Постановка задачи

Разрабатываемое программное обеспечение (ПО) предназначено для повышения эффективности и качества работы всех штатных сотрудников клуба федерации ездового спорта, а также для удобства и оперативности деятельности членов клуба. Система должна хранить и обрабатывать информацию о членах клуба, их собак, а также спортивных достижениях. Система ориентирована на конечного пользователя, не обладающего высокой квалификацией в области вычислительной техники. Поэтому

автоматизированная информационная система должна обладать простым, удобным и легко осваиваемым интерфейсом. Соответственно должна быть разработана база данных, которая будет обеспечивать надежное хранение и доступ к информации.

Предполагаемой целевой аудиторией будут являться круг лиц, заинтересованных в получении оперативно информации касающейся деятельности спортивно-кинологического клуба

Категории пользователей информационной системы приведены в таблице 1.

Таблица 1 – Описание категорий пользователей

Категория пользователей	Наименование приложения (подсистемы)	Фрагмент структуры ПрОбл	Информационные задачи
Председатель клуба	Подсистема Председатель	Председатель члены клуба, членские взносы	– Регистрировать новых членов – Отслеживать уплату членских взносов – Управление соревнованиями
Кинолог	Приложение Кинолог	Кинолог, собаки, медицинские паспорта, награды, соревнования	– Учет собак клуба – Фиксирование наград собак – Формирование медицинских паспортов собак – Редактировать информацию о участии собак в заездах
Гость	Начальная форма	Клуб. контакты	– Просмотр информации о клубе – Просмотр контактов

После рассмотрения категорий пользователей и их информационных задач необходимо выделить бизнес-процессы в предметной области.

1.4 Анализ бизнес-процессов предметной области

Бизнес-процессы подсистемы председатель клуба представлены в таблице 2.

Таблица 2 – Бизнес-процессы подсистемы председатель клуба

Бизнес-процесс	Исполнитель	Входные данные		Выходные данные		Ограничения
		Поставщик	Содержание	Потреби-тель	Содержание	
Регистрировать новых членов	Председатель клуба	Члены клуба	Сведения о членах клуба	Члены клуба, администрация	Список членов клуба	Члены клуба должны владеть собаками
Отслеживать уплату членских взносов	Председатель клуба	Члены клуба	Сведения о уплате членских взносов	Администрация	Список членов клуба имеющих/неимеющих задолженности по членским взносам	Членские взносы уплачиваются в фиксированном размере, за заданный период времени
Управление соревнованиями	Председатель клуба, Кинолог	Кинолог	Сведения соревнованиях ,сведения о составах заездов	Члены клуба	Список соревнований, список заездов	Заезды регулируются типами упряжки и типом соревнования

Бизнес-процессы подсистемы кинолога представлены в таблице 3.

Таблица 3 – Бизнес-процессы приложения подсистемы кинолога

Бизнес-процесс	Исполнитель	Входные данные		Выходные данные		Ограни-чения
		Поставщик	Содержание	Потребитель	Содержа-ние	
1	2	3	4	5	6	7
Учет собак членов клуба	Кинолог	Члены клуба	Данные о собаках	Члены клуба, администрация	Обновление сведений о собаках членов клуба	Собаки имеет ограничение по продолжительности спортивной формы
Фиксирование наград собак членов клуба	Кинолог	Председатель	Результаты соревнования проводимых в клубе	Члены клуба, администрация	Обновление сведений о наградах и титулах собак по результатам соревнований	Учет наград только клубных соревнований

Продолжение таблицы 3

1	2	3	4	5	6	7
Формирование медицинских паспортов собак	Кинолог	Члены клуба, ветстанция	Сведения о имеющихся прививках и травмах и болезнях собак	Члены клуба, администрация	Обновление сведений о медицинском, состоянии собак	Сведения имеют фиксированную форму
Управление соревнованиями	Председатель клуба, Кинолог	Кинолог	Сведения соревнованиях ,сведения о составах заездов	Члены клуба	Список соревнований, список заездов	Заезды регулируются типами упряжки и типом соревнования

Бизнес-процессы гостя клуба представлены в таблице 4.

Таблица 4 – Бизнес-процессы гостя

Бизнес-процесс	Исполнитель	Входные данные		Выходные данные		Ограничения
		Поставщик	Содержание	Потреби-тель	Содержание	
Просмотр информации о клубе	Гость			Гость	Сведения о целях и задачах клуба	–
Просмотр информации о контактах	Гость			Гость	Контактная информация о клубе	–

Выделенные бизнес-процессы в дальнейшем будут использоваться при проектировании и реализации серверной части разрабатываемого программного продукта.

Для того чтобы наиболее полно представлять себе работу данной системы следует рассмотреть ее в понятиях, используемых для описания предметной области, ее свойств и характеристиках, и законов протекания процессов в ней.

Типы сущностей предметной области приведены в таблице 5.

Таблица 5 – Типы сущностей предметной области

Наименование	Описание	Категория
Владелец	Список членов клуба	Стержневая
Оплата	Список с информацией о уплаченных взносах	Стержневая
Собака	Список с собак членов клуба	Стержневая
Соревнование	Список соревнований в клубе	Стержневая
Награды собак	Список наград и титулов собак	Стержневая
Медицинский паспорт собаки	Список медицинских данных у собак	Стержневая
Заезды	Составы заездов на соревнованиях	Стержневая
Породы	Список пород ездовых собак	Характеристическая
Титулы	Перечень титулов и наград собак	Характеристическая
Тип соревнований	Список типов соревнований	Характеристическая
Тип саней	Перечисление типов саней	Характеристическая
Тип упряжки	Перечисления типов упряжки	Характеристическая

Типы связей между типами сущностей предметной области приведены в таблице 6.

Таблица 6 – Типы связей между типами сущностей предметной области

Тип сущности 1	**Тип сущности 2**	**Описание**	**Мощность**	**Обязательность**
1	2	3	4	5
Владелец	Собака	Собака принадлежит владельцу Владелец имеет собак	1:M	Обязательная (у собаки должен быть владелец)
Владелец	Оплата	Список содержит оплаченные владельцами взносы оплаченный взнос учтен	1:1	Обязательная (содержит и оплаченные и неоплаченные взносы)

Продолжение таблицы 6

1	2	3	4	5
Собака	Медицинский паспорт собаки	Собака может иметь медицинский паспорт Паспорт принадлежит собаке	1:1	Обязательная (медицинский паспорт содержится в списке собак)
Порода	Собака	Собака имеет содержит породу Породы собак содержатся в списке о собаках	1:1	Обязательная (собака имеет породу)
Собака	Участие в заезде	Собаку определяют в заезд / заезды состоять из разных сочетаний собак	1: М	Обязательная (не все собаки участвуют в заездах)
Собака	Награды	У собаки есть награды/ награда принадлежит собаке	1:М	Обязательная (у собаки может не быть наград)
Награды	Титулы	У награды есть разные (титулы)статусы//титул признак награды	1:М	Обязательная (у награды есть определённый титул)
Награды	Соревнования	награды результат соревнования/ у соревнования есть награды есть издания	М:1	Обязательная (соревнования можно без награды)
Тип	Соревнование	У каждого соревнования есть тип/каждому типу соответствуют свои соревнования	1:М	Обязательная (у любого соревнования вседа есть тип)
Соревнование	Участие в заезде	каждое соревнование состоит из заездов/ заезд проходит в рамках соревнования	1:М	Обязательная (у любого соревнования есть заезды)
Тип саней	Участие в заезде	В заезде определенные типы саней/тип саней в одном заезде	М:1	Обязательная (у любого заезда есть определенный тип саней)

Окончание таблицы 6

1	2	3	4	5
Тип упряжки	Участие в заезде	В заезде определенные типы упряжек/тип упряжек в одном заезде	M:1	Обязательная (у любого заезда есть определенный тип упряжек)
Медицинский паспорт собаки	Тип записи	В паспорте разные типы записей/тип записи находится в паспорте	1:M	Необязательная (в любом паспорте есть определенные типы записи)

Спецификация сущности «Владельцы» приведена в таблице 7.

Таблица 7 – Спецификация сущности «Владельцы»

Наименование	Описание	Тип данных	Размер-ность	Область допусти-мых значений	Возможность значения Null	Роль	Пример
ID_книжки владельца	Номер книжки владельца	Целый	6	000001-999999	нет	PK	12
ФИО	ФИО владельца	Символьный	30	“А-я”	нет		Иванов Иван Иванович
Адрес	Адрес владельца	Символьный	40	“А-я”, «0-9»	нет		Ул. Московская 34 12
Год рождения	Год рождения владельца	Целый	4	0001-9999	нет		1995
Год вступления	Дата вступления в клуб	Целый	4	0001-9999	нет		2015

Спецификация сущности «Оплата» приведена в таблице 8.

Таблица 8 – Спецификация сущности «Оплата»

Наименование	Описание	Тип данных	Размерность	Область допустимых значений	Возможность значения Null	Роль	Пример
1	2	3	4	5	6	7	8
ID_книжки владельца	Номер книжки владельца	Целый	6	000001-999999	нет	FK	12

Продолжение таблицы 8

1	2	3	4	5	6	7	8
ID_счета	Номер счета оплаты	Целый	12	000000000001-999999999999999	нет		0000000050
Дата	Дата оплаты	Дата	8	-	нет		16.11.2020
Период	Период оплаты	Симво-льный	10	“А-я”	Нет		Март
Сумма	Сумма оплаты	Целый	4	1-1300	нет		1000
Оплачен /Неоплачен	Признак оплаты	Логичекий	2	Да-нет			да

Спецификация сущности «Собака» приведена в таблице 9.

Таблица 9 – Спецификация сущности «Собака»

Наименование	Описание	Тип данных	Размер-ность	Область допустимых значений	Возмож-ность значения Null	Роль	Пример
ID_книжки владельца	Номер книжки владельца	Целый	6	000001-999999	нет	FK	12
ID_клейма собаки	Номер клейма собаки	Целый	6	1-999999	Нет	PK	24
ID_породы	Номер породы собаки	Целый	1	0-9	Нет	FK	99
Заводчик	Фамилия заводчика	Символьный	15	“А-я”	есть		Злобин
ID_ Паспорта	Номер медицинского паспорта	Целый	6	1-999999	Нет	FK	24
Кличка	Кличка собаки	Симво-льный	25	“А-я”	Нет		Гредди Файст
Год рождения	Год рождения собаки	Целый	4	0001-9999	нет		2015

Спецификация сущности «Паспорт собаки» приведена в таблице 10.

Таблица 10 – Спецификация сущности «Паспорт собаки»

Наименование	Описание	Тип данных	Размер-ность	Область допустимых значений	Возмож-ность значения Null	Роль	Пример
ID_ Паспорта	Номер медицинского паспорта	Целый	6	1-999999	Нет	PK	24
ID_записи	Номер записи	Целый	2	1-99	Нет		24
ID_Тип записи	Номер типа записи	Целый	2	1-99	Нет	FK	11
Дата записи	Дата записи	Дата	8	-	нет		16.11.2020

Спецификация сущности «Тип записи» приведена в таблице 11.

Таблица 11 – Спецификация сущности «Тип записи»

Наименование	Описание	Тип данных	Размерность	Область допустимых значений	Возмож-ность значения Null	Роль	Пример
ID_Тип записи	Номер типа записи	Целый	2	1-99	Нет	PK	11
Запись	Значение записи	символьный	40	“А-я”	Нет		Прививка бещенст-во

Спецификация сущности «Породы» приведена в таблице 12.

Таблица 12 – Спецификация сущности «Породы»

Наименование	Описание	Тип данных	Размер-ность	Область допустимых значений	Возмож-ность значения Null	Роль	Пример
ID_породы	Номер породы собаки	Целый	1	0-9	Нет	PK	99
Название	Название Породы собаки	символьный	15	“А-я”	Нет		Маламут

Спецификация сущности «Награды собак» приведена в таблице 13.

Таблица 13 – Спецификация сущности «Награды собак»

Наименование	Описание	Тип данных	Размерность	Область допустимых значений	Возможность значения Null	Роль	Пример
ID_клейма собаки	Номер клейма собаки	Целый	6	1-999999	Нет	FK	24
ID_соревнований	Номер соревнования	Целый	1	1-9	Нет	FK	2
ID_типа соревнования	Номер типа соревнования	Целый	1	1-99	Нет	FK	2
ID_титула	Номер титула	Целый	1	1-9	Нет	FK	1
Награда	Награда соревнований	символьный	25	“А-я”	Нет		Диплом хорошо

Спецификация сущности «Титул» приведена в таблице 14.

Таблица 14 – Спецификация сущности «Титул»

Наименование	Описание	Тип данных	Размерность	Область допустимых значений	Возможность значения Null	Роль	Пример
ID_титула	Номер титула	Целый	1	1-9	Нет	PK	1
Титул	Название титула собаки	Символьный	20	“А-я”	Нет		Чемпион породы

Спецификация сущности «Соревнования» приведена в таблице 15.

Таблица 15 – Спецификация сущности «Соревнования»

Наименование	Описание	Тип данных	Размерность	Область допустимых значений	Возможность значения Null	Роль	Пример
1	2	3	4	5	6	7	8
ID_соревнований	Номер соревнования	Целый	1	1-99	Нет	PK	2

Продолжение таблицы 15

1	2	3	4	5	6	7	8
Дата	Дата проведения	Дата	8	-	нет		16.11.2020
Место	Место проведения	20	“А-я”	Нет	Нет		
ID_типа соревнования	Номер типа соревнования	Целый	1	1-99	Нет	FK	2

Спецификация сущности «Тип соревнования» приведена в таблице 16.

Таблица 16 – Спецификация сущности «Тип соревнования»

Наименование	Описание	Тип данных	Размерность	Область допустимых значений	Возмож-ность значения Null	Роль	Пример
ID_типа соревнований	Номер соревнования	Целый	2	1-99	Нет	PK	2
Название	Название соревнований	Символь-ный	20	“А-я”	Нет		Зимний митап

Спецификация сущности «Участие в заездах» приведена в таблице 17.

Таблица 17 – Спецификация сущности «Участие в заездах»

Наименование	Описание	Тип данных	Размерность	Область допустимых значений	Возмож-ность значения Null	Роль	Пример
ID_соревнований	Номер соревнования	Целый	1	1-9	Нет	FK	2
ID_клейма собаки	Номер клейма собаки	Целый	6	1-999999	Нет	PK	24
ID_саней	Номер типа саней	Целый	1	1-9	Нет	FK	8
ID_упряжки	Номер типа упряжки	Целый	1	1-9	Нет	FK	1
ID_ типа соревнований	Номер типа соревнований	Целый	1	1-9	Нет	FK	1

Спецификация сущности «Тип саней» приведена в таблице 18.

Таблица 18 – Спецификация сущности «Тип саней»

Наименование	Описание	Тип данных	Размерность	Область допустимых значений	Возмож-ность значения Null	Роль	Пример
ID_саней	Номер типа саней	Целый	1	1-9	Нет	PK	8
Название	Название саней	Символь-ный	20	“А-я”	Нет		Нарты
Вес груза	Грузоподь емность	Целый	6	1-999999	Нет	FK	24

Спецификация сущности «Тип упряжки» приведена в таблице 19.

Таблица 19 – Спецификация сущности «Тип упряжки»

Наименование	Описание	Тип данных	Размерность	Область допустимых значений	Возмож-ность значения Null	Роль	Пример
ID_упряжки	Номер типа упряжки	Целый	1	1-9	Нет	PK	1
Количество собак	Название п	Целый	2	1-12	Нет		4

Концептуальная модель предметной области, на уровне «сущность – связь» представлена на рисунке 6.

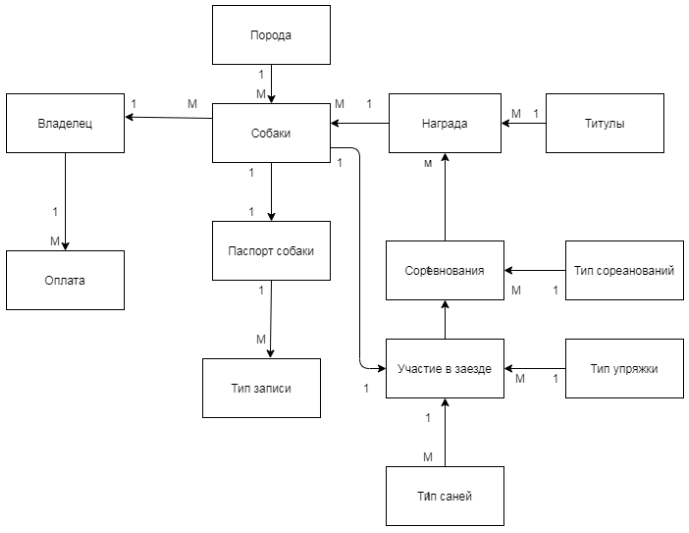


Рисунок 6 – Концептуальная модель уровня «сущность – связь»

В результате анализа бизнес-процессов были выделены основные сущности предметной области и построена концептуальная модель предметной области.

1.5 Анализ требований

1.5.1 Анализ функциональных требований

В данной системы существует три типа пользователей: председатель клуба, кинолог и гость.

Основными функциями председателя клуба являются:

- авторизация в системе;
- просмотр информации о владельцах;
- просмотр информации о оплате членских взносов;
- просмотр информации о заездах;
- просмотр информации о соревнованиях;
- добавление новых владельцев;
- удаление владельцев;
- редактирование информации о владельцах;
- поиск владельцев;
- учет членских взносов;
- добавление информации о членских взносах;
- изменение информации о членских взносах;
- поиск информации о членских взносах;
- добавление данных о соревнованиях.

Основными функциями кинолога являются:

- авторизация в системе;
- просмотр списка собак;
- добавление собак в список;
- удаление собак из списка;

– редактирование данных о собаке;

– добавление данных о наградах собак;

– поиск данных о наградах собак;

– добавление записей в медицинские паспорта собак;

– поиск информации в медицинских паспортах собак;

– просмотр информации о соревнованиях;

– просмотр информации о заездах;

– добавление данных о заездах;

– редактирование данных о заездах.

Основными функциями гостя является:

– просмотр информации о клуб;

– просмотр информации о контактах.

Диаграмма вариантов использования для подсистемы председателя клуба приведена на рисунке 7.

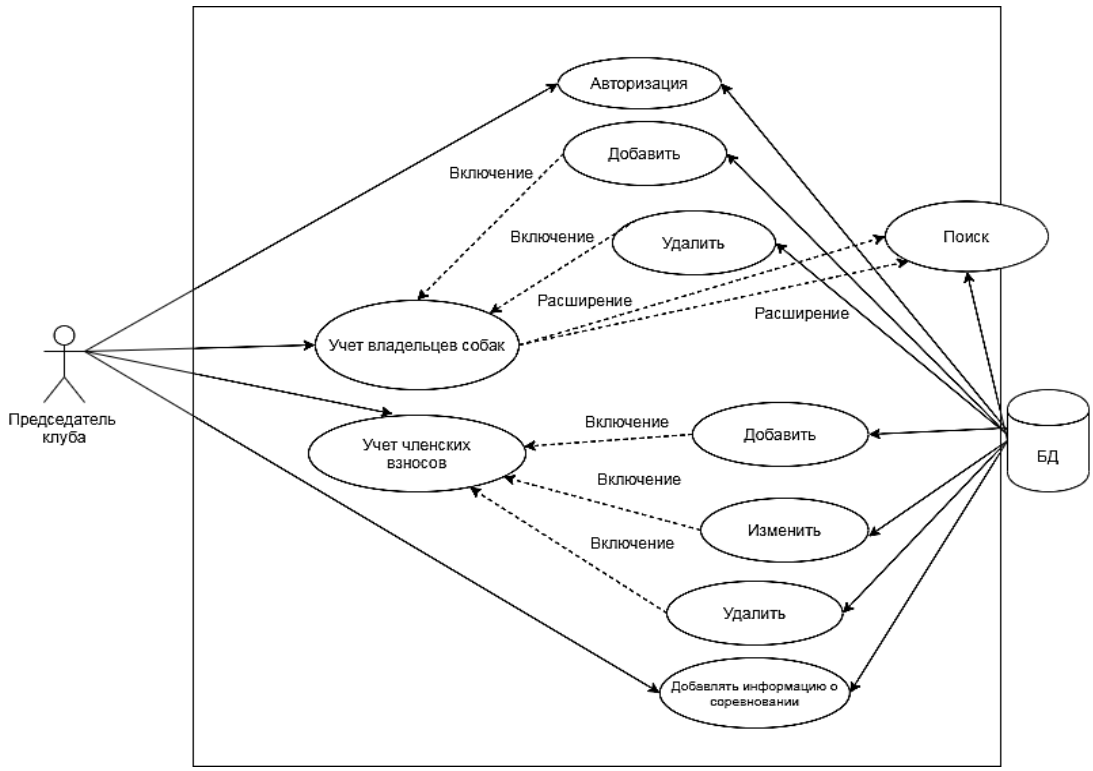


Рисунок 7 – Диаграмма вариантов использования подсистемы председателя клуба

Диаграмма вариантов использования для подсистемы кинолога приведена на рисунке 8.

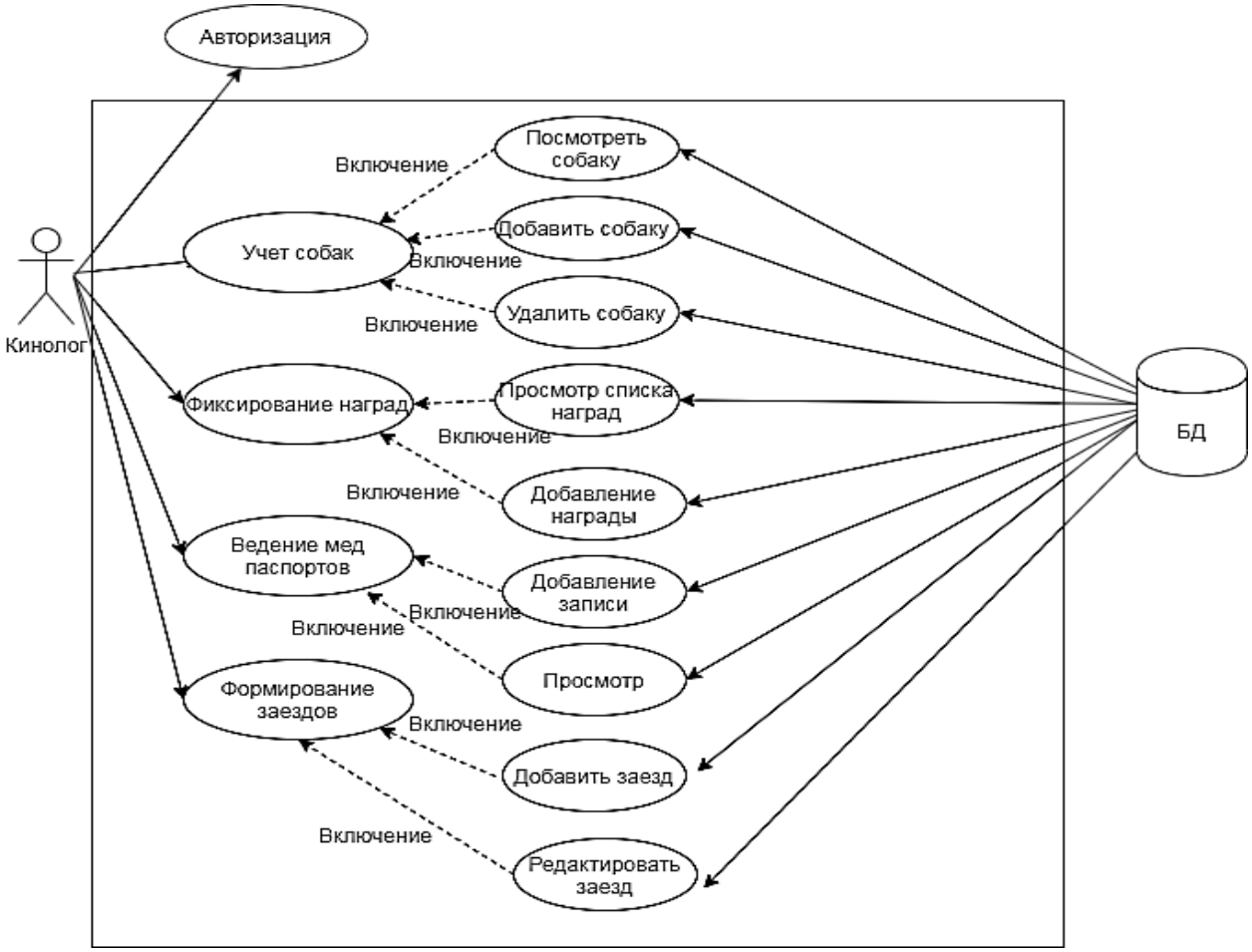


Рисунок 8 – Диаграмма вариантов использования подсистемы кинолога

Диаграмма вариантов использования для гостя приведена на рисунке 9.

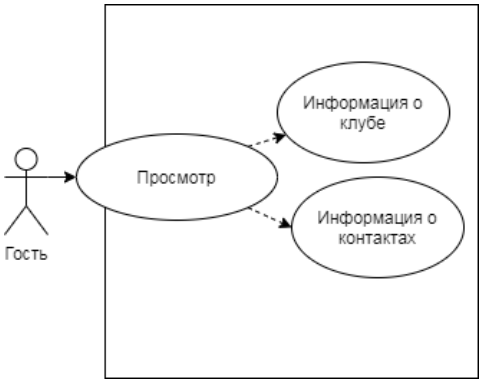


Рисунок 9 – Информация для гостя

Далее будет рассмотрены спецификации основных прецедентов [8].

Спецификация прецедента «Добавить владельца» представлена на рисунке 10.

Прецедент: Учет владельцев собак
ID: 1
Краткое описание: Председатель ведет учет владельцев собак клуба
Главные актеры: Председатель
Второстепенные актеры: СУБД
Предусловия: Председатель выполняет просмотр списка владельцев
Основной поток: 1. Include (Добавить владельца) 2. Include (Найти владельца) 3. Председатель подтверждает добавления владельца 4. Система выводит сообщение об успешном добавлении владельца.
Постусловия: Система обновляет список владельцев собак
Альтернативные потоки: Нет

Рисунок 10 – Спецификация прецедента «Добавить владельца»

Спецификация прецедента «Учет членских взносов» представлена на рисунке 11.

Прецедент: Учет членских взносов
ID: 2
Краткое описание: Председатель отмечает факт оплаты членских взносов
Главные актеры: Председатель
Второстепенные актеры: СУБД
Предусловия: Председатель выбирает функцию «Учет членских взносов».
Основной поток: 1. Include (Найти владельца) 1. Система выводит форму для поиска записи о владельце 2. Председатель заполняет данные о владельце 3. Подтверждает ввод данных 4. Система выводит сообщение об успехе. 5. Система добавляет отметку об уплате владельцем взносов в БД
Постусловия: Система обновляет список оплаченных взносов
Альтернативные потоки: Нет

Рисунок 11 – Спецификация прецедента «Учет членских взносов»

Спецификация прецедента «Добавление владельцев собак» представлена на рисунке 12.

Прецедент: Добавление владельцев собак
ID: 3
Краткое описание: Председатель добавляет владельца
Главные актеры: Председатель
Второстепенные актеры: СУБД
Предусловия: Председатель заходит на вкладку «Учет владельцев собак», выбирает функцию «Добавить ».
Основной поток: 1. Система выводит форму ввода данных о владельце 2. Заведующий библиотечного фонда заполняет данные о книге 3. Подтверждает ввод данных 4. Система выводит сообщение об успехе. 5. Система добавляет запись о владельце в БД
Постусловия: Система обновляет список владельцев собак
Альтернативные потоки: Нет

Рисунок 12 – Спецификация прецедента «Добавление владельцев собак»

Спецификация прецедента «Посмотреть владельцев» представлена на рисунке 13.

Прецедент: Посмотреть список владельцев
ID: 4
Краткое описание: Председатель просматривает списки членов клуба, владельцев собак
Главные актеры: Председатель
Второстепенные актеры: СУБД
Предусловия: Председатель заходит на вкладку «Учет владельцев собак», выбирает кнопку «Просмотр»
Основной поток: 1. Include (Найти владельца) 2. Система выводит список владельцев
Постусловия: Нет

Рисунок 13 – Спецификация прецедента «Посмотреть список владельцев»

Диаграмма деятельности для прецедента «Добавить собаку» представлена на рисунке 14.

Рисунок 14 – Диаграмма деятельности для прецедента «Добавить собаку»

Диаграмма деятельности для прецедента «Учет членских взносов» представлена на рисунке 15.

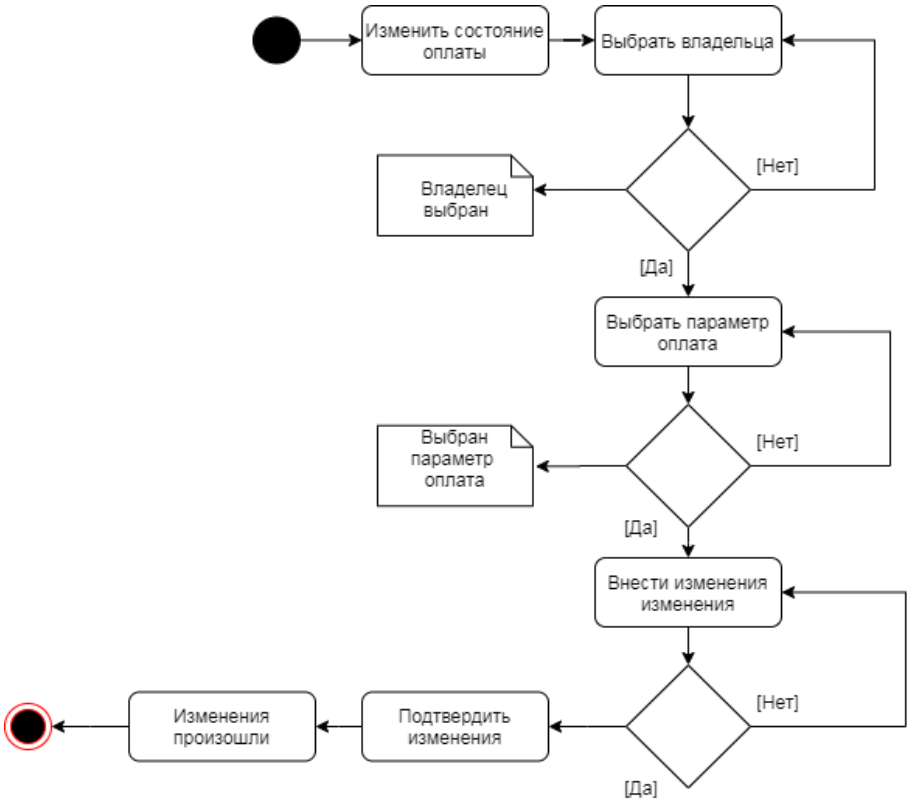


Рисунок 15 – Диаграмма деятельности для прецедента «Учет членских взносов» представлена на рисунке»

В результате были рассмотрены основные требования к функциональности системы.

1.5.2 Анализ нефункциональных требований

К нефункциональным требованиям относятся следующие критерии:

– интерфейсы пользователя;
– требование к производительности;
– атрибуты качества программного обеспечения (ПО);
– требования к безопасности;
– требование к архитектуре.

Программное обеспечение клуба федерации ездового спорта должно обладать удобным, простым и достаточно понятным для пользователя графическим интерфейсом. ПО должно обеспечивать стабильную работу при наличии стабильного соединения с сервером. Время ожидания данных от сервера не должно превышать 1 минуты.

Списки владельцев, собак, информация о собаках и соревнованиях должны автоматически обновляться. Когда в систему добавляется новый владелец или новая собака, информация о соревнованиях и заездах, то происходит добавление в базу данных и обновление во всех экземплярах клиента.

Доступ к клиентскому приложению должен осуществляться с помощью аутентификации пользователя. Председатель не должен получать доступ к функциям кинолога, но может просматривать результаты некоторых из них и наоборот.

В качестве пользовательского интерфейса было принято решение использовать оконный графический интерфейс, декстопного приложения, так как для него существует достаточное количество средств автоматизированного проектирования.

На основе анализа функциональных требований и анализа предметной области было принято решение использовать архитектуру приложения «клиент-сервер».

1.6 Выбор инструментальных средств

Для структурированного хранения информации о читателях и библиотечном фонде было принято решение использовать реляционную базу данных, в качестве систем управления базой данных (СУБД) – MS SQL Server, а в качестве информационно-логического языка баз данных – SQL. Выбор обусловлен тем, что SQL – стандартный язык для работы с базами данных, который получил достаточно широкое распространение. Почти все разработчики СУБД в настоящее время конструируют свои продукты с использованием языка SQL либо с SQL-интерфейсом.

Проектирование логической и физической модели информационной системы выполняется при помощи case-средства ERWin Data Modeler r7.

Интегрированная среда разработки – Microsoft Visual Studio 19.

Для планирования, распределения ресурсов по задачам и отслеживания прогресса в анализе объемов работ выбрано приложение Microsoft Project 19. Выбор обусловлен распространённостью программного продукта и тем, что Microsoft Project разработан, с целью помочь менеджеру проекта в планировании, распределении ресурсов по задачам, отслеживании прогресса и анализе объемов работ.

1.7 Системные программные средства

Для создания автоматизированной системы была выбрана настройка Windows Form – интерфейс программирования оконных приложений, отвечающий за графический интерфейс пользователя и являющийся частью системы Microsoft .NET Framework.

В качестве языка программирования выбран C#. Выбор обусловлен тем, что данный язык обладает огромной библиотекой классов, позволяющей

удобно проектировать, как графический интерфейс, так и взаимодействовать с базой данных. Также C# постоянно развивается и совершенствуется, добавляются новые функции [9].

2 Проектирование информационной системы клуба федерации ездового спорта

2.1 Базовая архитектура системы

Рассматриваемая система имеет двухуровневую архитектуру клиент-сервер. Архитектура автоматизированной информационной системы представлена на рисунке 16.

Рисунок 16 – Схема архитектуры «клиент-сервер»

В данной архитектуре присутствуют три главных компонента:

– база данных, в которой хранятся данные о деятельности клуба;

– сервер базы данных, в котором содержится бизнес логика;

– клиент, который является интерфейсом для взаимодействия между пользователем и серверной частью информационной системы.

2.2 Логическая модель базы данных

Логическая база данных создавалась в Erwin Data Modeler r7.

На основе результатов, полученных на этапе анализа предметной области, строится логическая модель базы данных с учетом задач каждого из пользователей информационной системы.

Получено 9 связанных между собой сущностей.

Логическая модель данных на уровне сущностей представлена на рисунке 17.

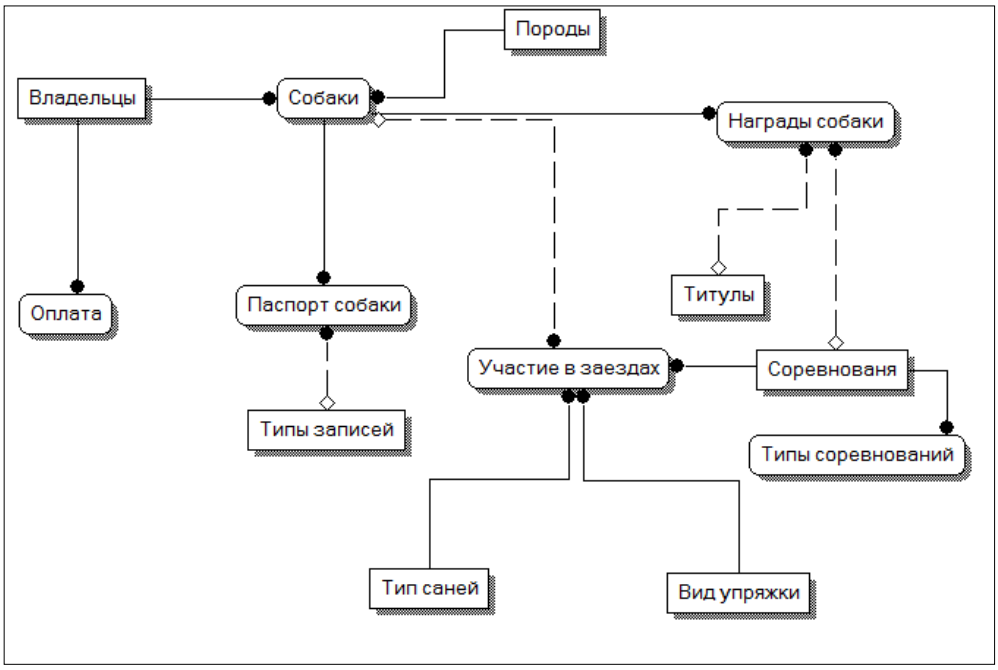


Рисунок 17 – Логическая модель базы данных на уровне сущностей

Логическая модель данных на уровне атрибутов представлена на рисунке 18.

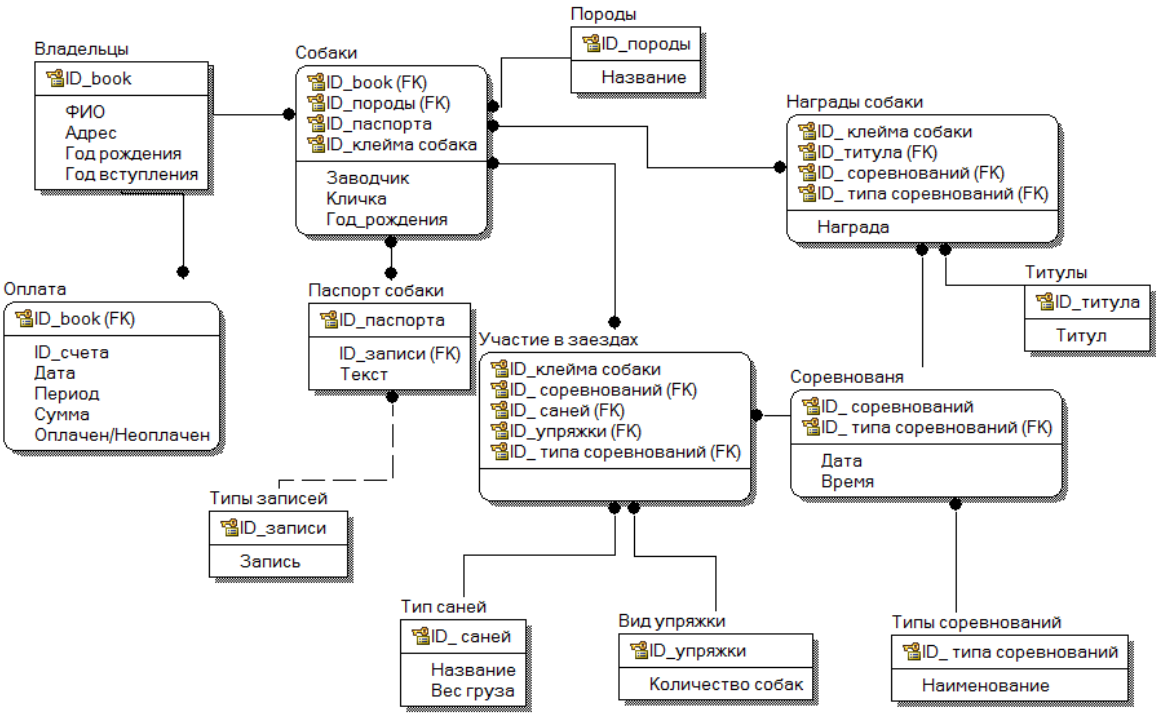


Рисунок 18 – Логическая модель базы данных на уровне атрибутов

Отчет, сгенерированный в среде Erwin Data Modeler r7 представлен на рисунке 19, 20.

Entity Name	Entity Logical Only	Entity Type	
Вид упряжки	No	Independent	
	Attribute Name	Attribute Is PK	Attribute Is FK
	ID_упряжки	Yes	No
	Количество собак	No	No
Владельцы	No	Independent	
	Attribute Name	Attribute Is PK	Attribute Is FK
	ID_book	Yes	No
	ФИО	No	No
	Адрес	No	No
	Год рождения	No	No
	Год вступления	No	No
Награды собаки	No	Dependent	
	Attribute Name	Attribute Is PK	Attribute Is FK
	ID_ клейма собаки	Yes	No
	ID_титула	Yes	Yes
	ID_ соревнований	Yes	Yes
	ID_ типа соревнований	Yes	Yes
	Награда	No	No
Оплата	No	Dependent	
	Attribute Name	Attribute Is PK	Attribute Is FK
	ID_book	Yes	Yes
	ID_счета	No	No
	Дата	No	No
	Период	No	No
	Сумма	No	No
	Оплачен/Неоплачен	No	No
Паспорт собаки	No	Independent	
	Attribute Name	Attribute Is PK	Attribute Is FK
	ID_паспорта	Yes	No
	ID_записи	No	Yes
	Текст	No	No
Породы	No	Independent	
	Attribute Name	Attribute Is PK	Attribute Is FK
	ID_породы	Yes	No
	Название	No	No
Собаки	No	Dependent	
	Attribute Name	Attribute Is PK	Attribute Is FK
	ID_book	Yes	Yes
	ID_породы	Yes	Yes
	ID_паспорта	Yes	No
	ID_клейма собака	Yes	No
	Заводчик	No	No
	Кличка	No	No
	Год_рожления	No	No
Соревнования	No	Dependent	
	Attribute Name	Attribute Is PK	Attribute Is FK
	ID_ соревнований	Yes	No
	ID_ типа соревнований	Yes	Yes

Рисунок 19 – Отчет по логической модели базы данных(часть 1)

	Дата	No	No
	Время	No	No
Тип саней	No	Independent	
	Attribute Name	Attribute Is PK	Attribute Is FK
	ID_ саней	Yes	No
	Название	No	No
	Вес груза	No	No
Типы записей	No	Independent	
	Attribute Name	Attribute Is PK	Attribute Is FK
	ID_записи	Yes	No
	Запись	No	No
Типы соревнований	No	Independent	
	Attribute Name	Attribute Is PK	Attribute Is FK
	ID_ типа соревнований	Yes	No
	Наименование	No	No
Титулы	No	Independent	
	Attribute Name	Attribute Is PK	Attribute Is FK
	ID_титула	Yes	No
	Титул	No	No
Участие в заездах	No	Dependent	
	Attribute Name	Attribute Is PK	Attribute Is FK
	ID_клейма собаки	Yes	No
	ID_ соревнований	Yes	Yes
	ID_ саней	Yes	Yes
	ID_упряжки	Yes	Yes
	ID типа соревнований	Yes	Yes

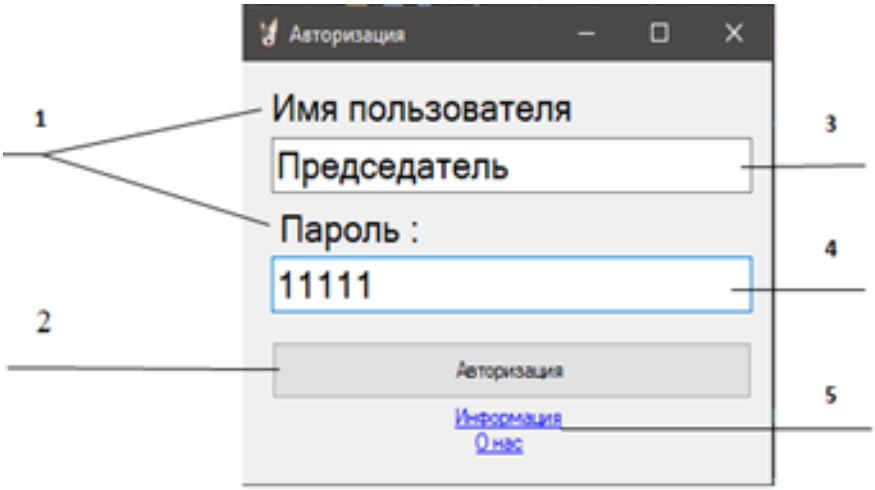
Рисунок 20 – Отчет по логической модели базы данных(часть 2)

В ходе дальнейшей работы будет реализован переход от логической модели базы данных к физической.

2.3 Проектирование пользовательского интерфейса

Исходя из диаграммы вариантов использований, был спроектирован графический пользовательский интерфейс, в котором реализовано разделение функций в зависимости от авторизированного пользователя.

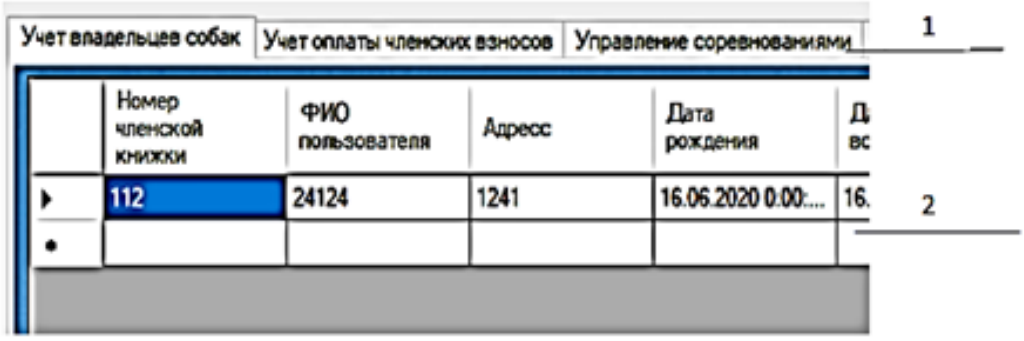
Интерфейс формы авторизации представлен на рисунке 21.



1 – надпись, компонент Lable;
2 – кнопка подтверждения ввода, компонент ButtonS;
3 – место ввода логина, компонент TextBox;
4 – место ввода пароля, компонент TextBox;
5 – гиперссылка для перехода на документ, компонент LincLabel.

Рисунок 21 – Интерфейс формы авторизации

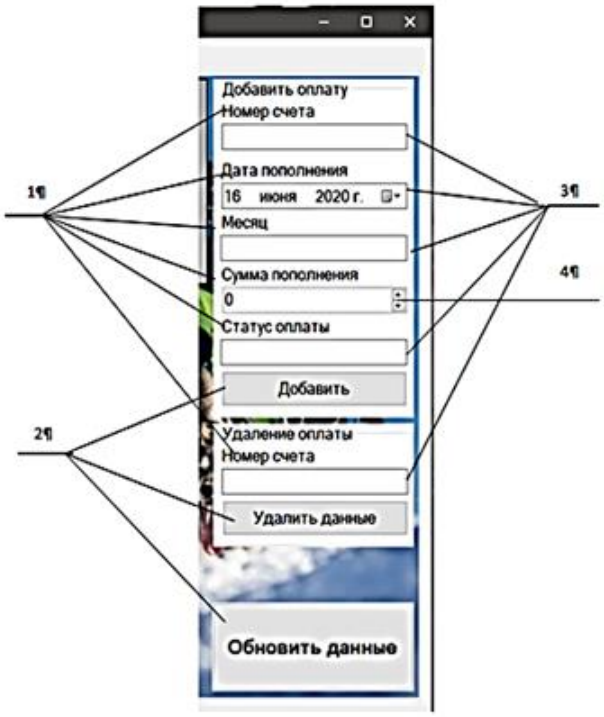
Интерфейс вкладок подсистемы председатель представлен на рисунке 22.



1 – вкладки с функционалом подсистемы, компонент TabPage;
2 – отображение таблиц базы данных, компонент DataGreedView.

Рисунок 22 – Интерфейс вкладок подсистемы председатель

Интерфейс панели управления на вкладках подсистемы председателя представлен на рисунке 23.



1 – надпись, компонент Lable;
2 – кнопка функциональная, компонент Button;
3 – место ввода данных, компонент TextBox;
4 – счетчик для установки значений, компонент NumericUpDown.

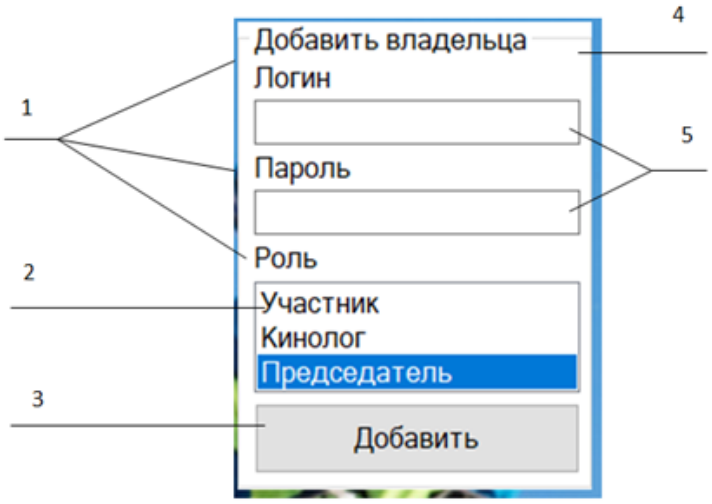
Рисунок 23 – Интерфейс панели управления на вкладках подсистемы председателя

Интерфейс поля ввода даты представлен на рисунке 24.

1 – надписи, компонент Lable;
2 – место даты, компонент TextBox;
3 – ввод дат, компонент Calendare.

Рисунок 24 – Интерфейс поля ввода даты

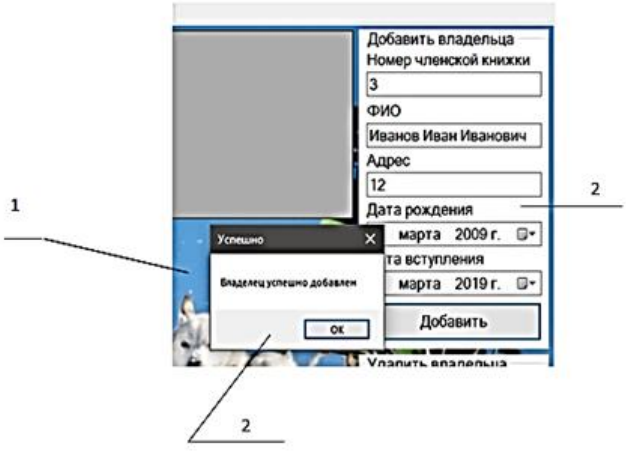
Интерфейс панели управлния пользователями на рисунке 25.



1 – надписи, компонент Lable;
2 – таблица подстановки, элемент ListBox;
3 – кнопка подтверждения действия, компонент Button;
4 – контейнер для компонентов, компонент Panell;
5 – ввод текстовых данных, компонент TextBox.

Рисунок 25 – Интерфейс панели управления пользователями

Интерфейс окна подсистемы Кинолог, после операции добавления представлен на рисунке 26.



1 – оконная форма, компонент Form;
2 – стандартное окно вывода, компонент MessageBox;
3 – контейнер для управляющих элементов, компонент Panel.

Рисунок 26 – Интерфейс окна после операции добавления

Фрагмент интерфейса окна кинолога, представлен на рисунке 27.

1 – надпись с фавиконом, компонент Form.Text;
2 – вкладки длс функций подсистемы «Кинолог», компонент TabControl;
3 – оконная форма, компонент Form.

Рисунок 27 – Фрагмент интерфейса окна кинолога

Фрагмент интерфейса вкладки Добавления собаки окна подсистемы кинолог представлен на рисунке 28.

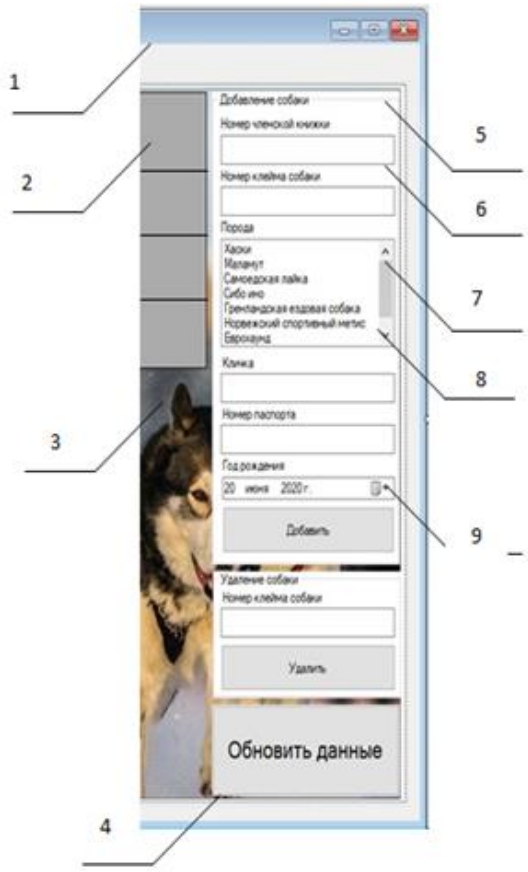


Рисунок 28 – Фрагмент интерфейса вкладки Добавления собаки окна подсистемы кинолог

Рассмотрим основные элементы фрагмента интрефейса вкладки «Добавление собаки», представленные на рисунке 28:

1 – оконная форма для подситемы «Кинолог», компонент Form;

2 – отображение табличных данных, компонент DataGridView;

3 – оформление окна, компонет Picture;

4 – кнопка обновления данных, компонет Button;

5 – пояснительные надписи, компонет Lable;

6 – элементы для ввода данных, компонет TextBox;

7 – полоса прокрутки, компонет ScroolBar;

8 – выпадающий список с таблицей подстановки, компонет ListBox;

9 – ввод даты, компонет DataTimePicer.

2.4 Проектирование форм отчетов

Так как система предназначена для учета оплаты членских взносов владельцев собак, формирования протоколов заездов на соревнования и фиксирование наград и титулов собак то необходимо предусмотреть возможность создания печатных форм для отчетов о деятельности клуба федерации ездового спорта, протоколов заездов, и ведомостей наград.

Отчет о владельцах, с указанием уплаты членских взносов рисунке 29.

ID_книжки владельца	ID_счета	Дата	Период	Суммам	Оплачен/ не оплачен
1	580000001	11.01.2020	декабрь	1200	да
2	580000002	14.01.2020	ноябрь	1100	да
3	580000003	17.01.2020	ноябрь	500	нет
4	580000004	20.01.2020	декабрь	1100	нет
5	580000005	23.01.2020	январь	1200	да

Рисунок 29 – Отчет о владельцах с указанием членских взносов

В отчете приведены основные сведения о владельцах с указанием номера членской книжки под которым каждый из владельцев зарегистрирован в клубе.

Макет формы протокола заезда на соревнованиях ездового спорта приведен на рисунке 30.

Протоколы заездов
2 этапа зимнего чемпионата MITAP
ездового спорта

Номер заезда	Тип саней	Тип упряжк	Номера собак	Номер по приходу в заезде
1	**нарты**	**4**	99	1
			113	
			1123	
			5	
1	**нарты**	**4**	88	2
			12	
			114	
			200	
2	**сани**	**2**	44	2
			140	1
2	**сани**	**2**	21	
			34	
3	**сани**	**2**	201	4
			202	

Рисунок 30 – Макет формы протокола соревнований

Далее приведены основные средние показатели работы клуба федерации ездового спорта:

Количество членов клуба;

Количество владельцев собак;

Количество собак, зарегистрированных на владельцев;

Среднее количество собак на одного владельца:

$$SR = \frac{VD}{V},$$

где SR – среднее количество собак на одного владельца;

VD – количество собак зарегистрированных на владельцев

V – количество владельцев собак;

Общая сумма денег в фонде клуба, за счет получение членских взносов

Количество соревнований, проводимых в рамках календаря клуба;

Количество собак призеров;

Удельное количество собак призеров соревнование к общему числу собак:

$$Ky = \frac{DP}{VD},$$

где DP – количество собак призеров;

VD – количество собак зарегистрированных на владельцев.

Форма отчета о работе клуба представлена на рисунке 31.

Отчет о работе клуба
за 2018-2019 год

Количество членов клуба	191;
Количество владельцев собак	170
Количество собак зарегистрированных на владельцев	311
Среднее количество собак на одного владельца	1,83
Общая сумма денег в фонде клуба , за счет получение членских взносов	15900 руб.
Количество соревнований в рамках календаря клуба	11
Количество собак призеров	49
Удельное количество собак призеров соревнование к общему числу собак	28%

Рисунок 31 – Форма отчета о работе клуба

Форма страницы записи медицинского паспорта собаки на рисунке 32.

Кличка: Суоми Д/р: **11.08.15** Порода: метис Пол сука № клейма собаки **001208956**

№ записи	Запись	Дата	Примечания
1	Прививка ОКД-18	11.11.17	Ауто/имун ответ получен
2	Плановая вязка	01.01.18	По плану, кобель Атлас
3	Щенность	05.05.18	В помете 6 щенков
4	Травма сустава лрб	01.10.19	
5	Прививка Бещенство	11.12.19	
6	Прививка чумка	14.02.20	

Рисунок 32 – Форма страницы записи медицинского паспорта

Согласие на обработку персональных данных представлено на рисунке 33 [11].

Согласие на обработку персональных данных

Я,

(Фамилия, Имя, Отчество)

Паспорт (серия, номер)

Кем, когда выдан _____

подтверждаю, что я ознакомлен и полностью согласен с условиями оказания мне библиотечных услуг, я согласен с тем, что библиотека может отказать мне в обслуживании в случае их нарушения. Также даю свое согласие на обработку моих персональных данных, указанных в настоящей регистрационной карточке. Данное согласие действует до моего прямого отказа от пользования услугами библиотеки выраженного мною лично в устной или письменной форме, либо до истечения двухлетнего срока с момента последнего уточнения моих данных (перерегистрации).

_____ _____ (_____)

Дата Подпись (Расшифровка подписи)

Рисунок 33 – Согласие на обработку персональных данных

В результате были спроектированы очновные формы для отчетов о деятельности клуба федерации ездового спорта.

2.5 Технология разработки

2.5.1 Шаблон проектирования «Адаптер»

Для конструирования программного обеспечения был выбран шаблон «Адаптер».

Адаптер – это структурный шаблон проектирования, предназначенный для преобразования интерфейса одного класса в интерфейс другого. Благодаря реализации данного паттерна можно использовать вместе классы с несовместимыми интерфейсами [12].

Диаграмма классов шаблона «Адаптер» представлена на рисунке 34.

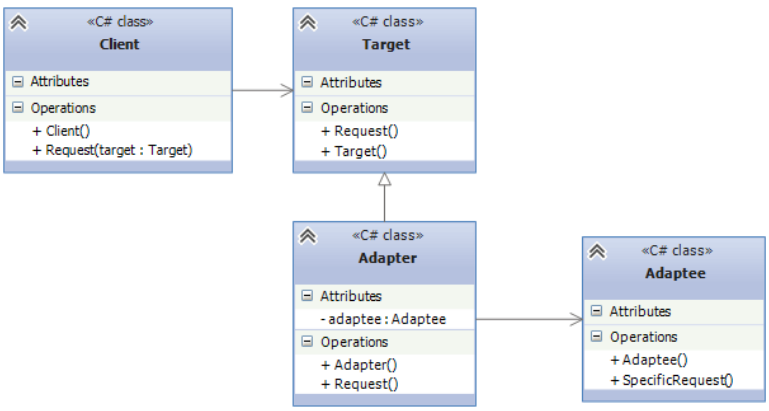


Рисунок 34 – Диаграмма классов шаблона «Адаптер»

На диаграмме изображены следующие классы:

– Target: представляет объекты, которые используются клиентом;

– Client: использует объекты Target для реализации своих задач;

– Adaptee: представляет адаптируемый класс, который мы хотели бы использовать у клиента вместо объектов Target;

– Adapter: собственно, адаптер, который позволяет работать с объектами Adaptee как с объектами Target.

То есть клиент ничего не знает об Adaptee, он знает и использует только объекты Target. И благодаря адаптеру можно на клиенте использовать объекты Adaptee как Target.

2.6 Проектирование программных средств

2.6.1 Общее проектирование системы

На основе анализа предметной области и паттерна «Адаптер» была разработана диаграмма классов, представленная на рисунке 35.

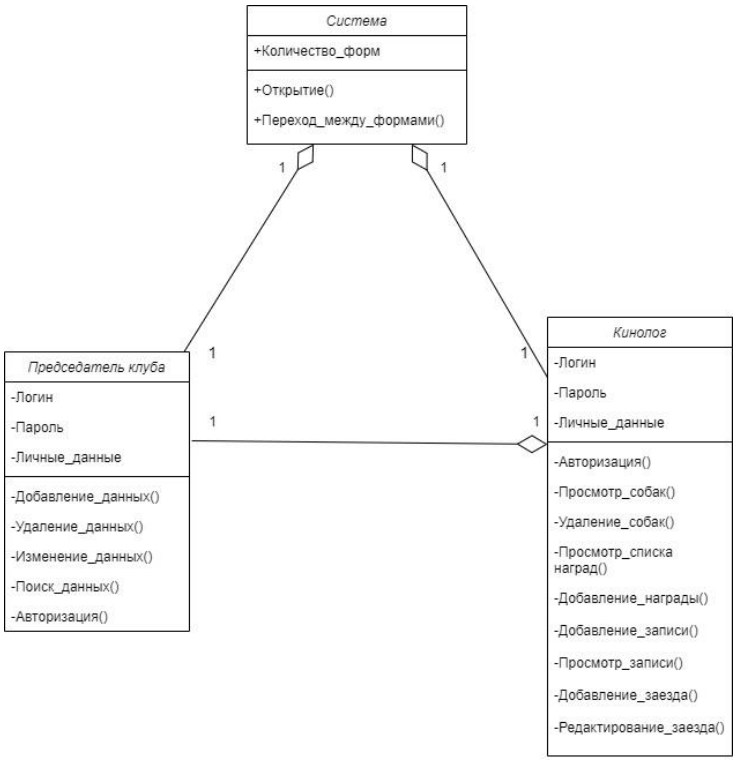


Рисунок 35 – Диаграмма классов приложения «Информационная система клуба Федерации ездового спорта»

На диаграмме изображены следующие классы:

– Система – главный класс приложения;

– Председатель клуба – класс подсистемы Председатель;

– Кинолог – класс подсистемы Кинолог.

Развернутая диаграмма классов артефактов таблиц, ER диаграмма представлена на рисунке 36.

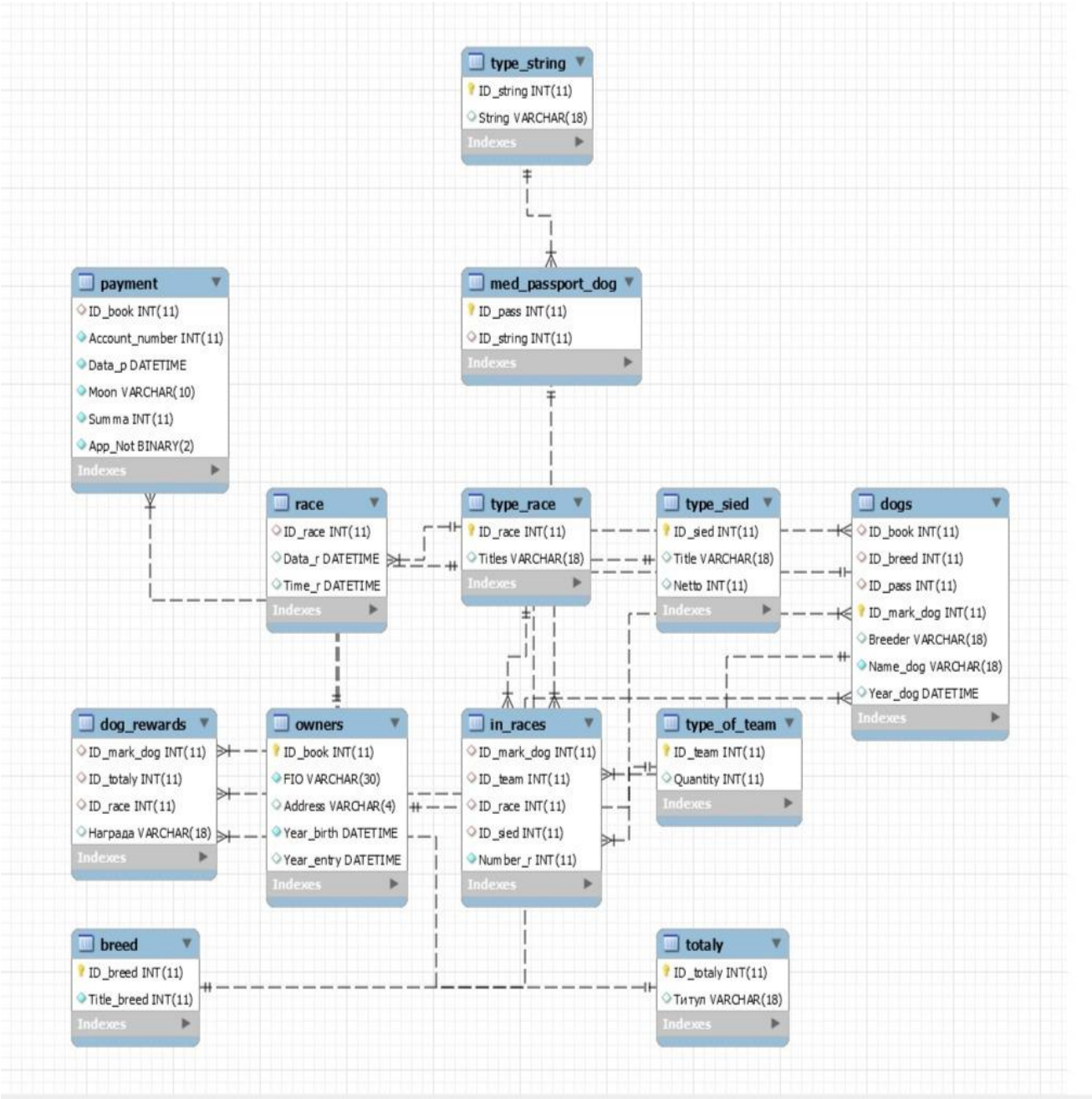


Рисунок 36 – Диаграмма классов приложения

На диаграмме изображены следующие классы:

– payment – класс обрабатывающий данные об оплате;

– med_passport_dog – класс обработки информации о медицинских паспортах собак и записей в них;

– rase – класс обрабатывающий добавление соревнований.

– type_rase– класс содержащий типы проводимых соревнований;

– dogs– класс обработки и хранения данных о собаках клуба;

– dogs_rewards– класс формирования данных о владельцах собак;

– owners – обработчик события загрузки окна;

– in_rase – класс формирующий состав собак в заездах на соревнованиях;

– type_of_team – класс хранения типов соревнований;

– bread – класс для типов саней участников заездов:

– totaly – класс обработка и фиксирование наград и титулов собак участников соревнований.

3 Реализация информационной системы клуба федерации ездового спорта

3.1 Физическая модель базы данных

Физическая модель строится на основе логической модели данных и учитывает особенности целевой СУБД. В качестве целевой СУБД выбран SQL Server. В среде ERWin можно с легкостью перейти от логической модели к физической.

Физическая модель базы данных представлена на рисунке 37.

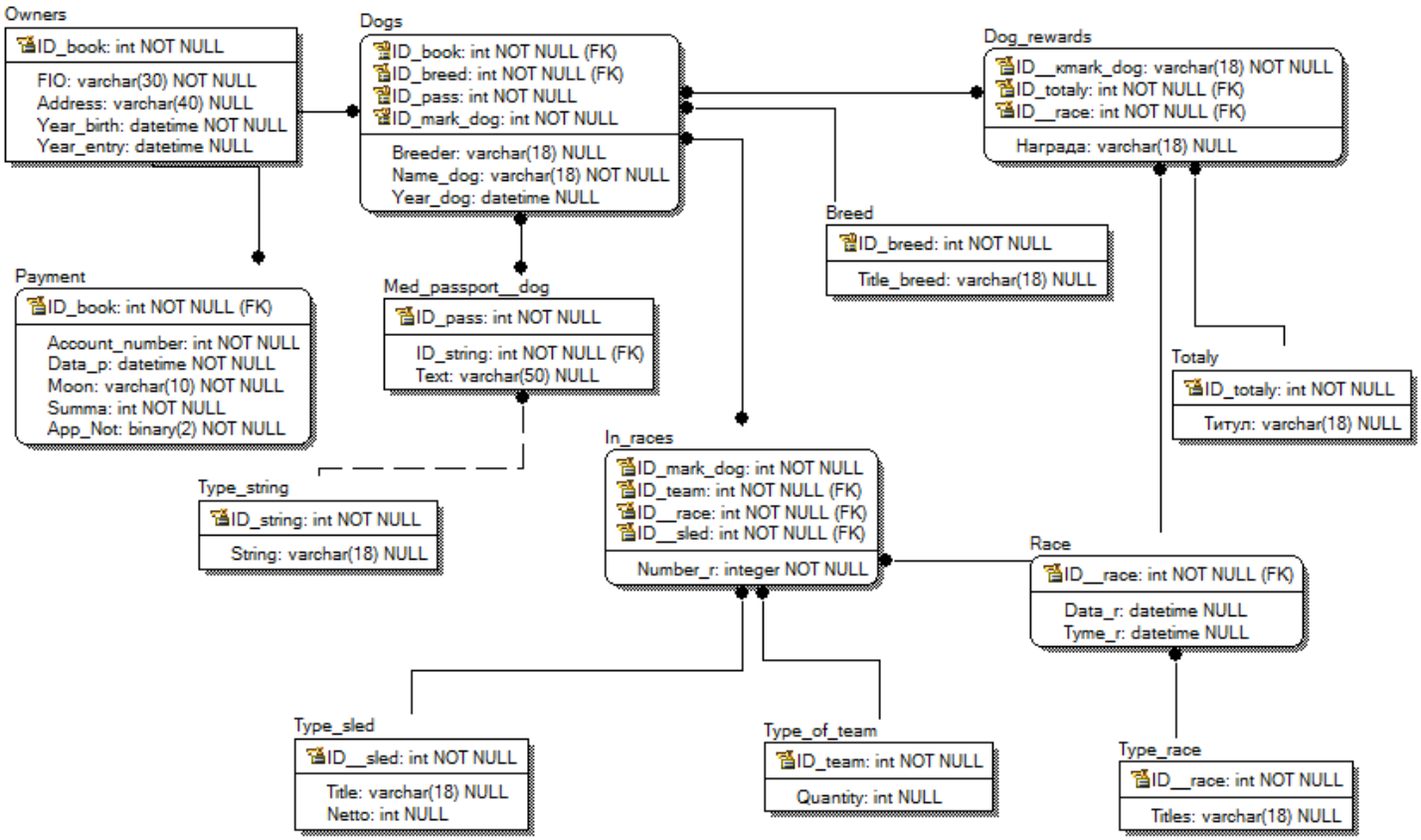


Рисунок 37 – Физическая модель базы данных

Сгенерированный скрипт создания базы данных в ERWin Data Modeler был выполнен в СУБД MS SQL Server. В результате была получена схема базы данных, представленная на рисунке 38.

Листинг скрипта для создания базы данных приведен в приложении В.

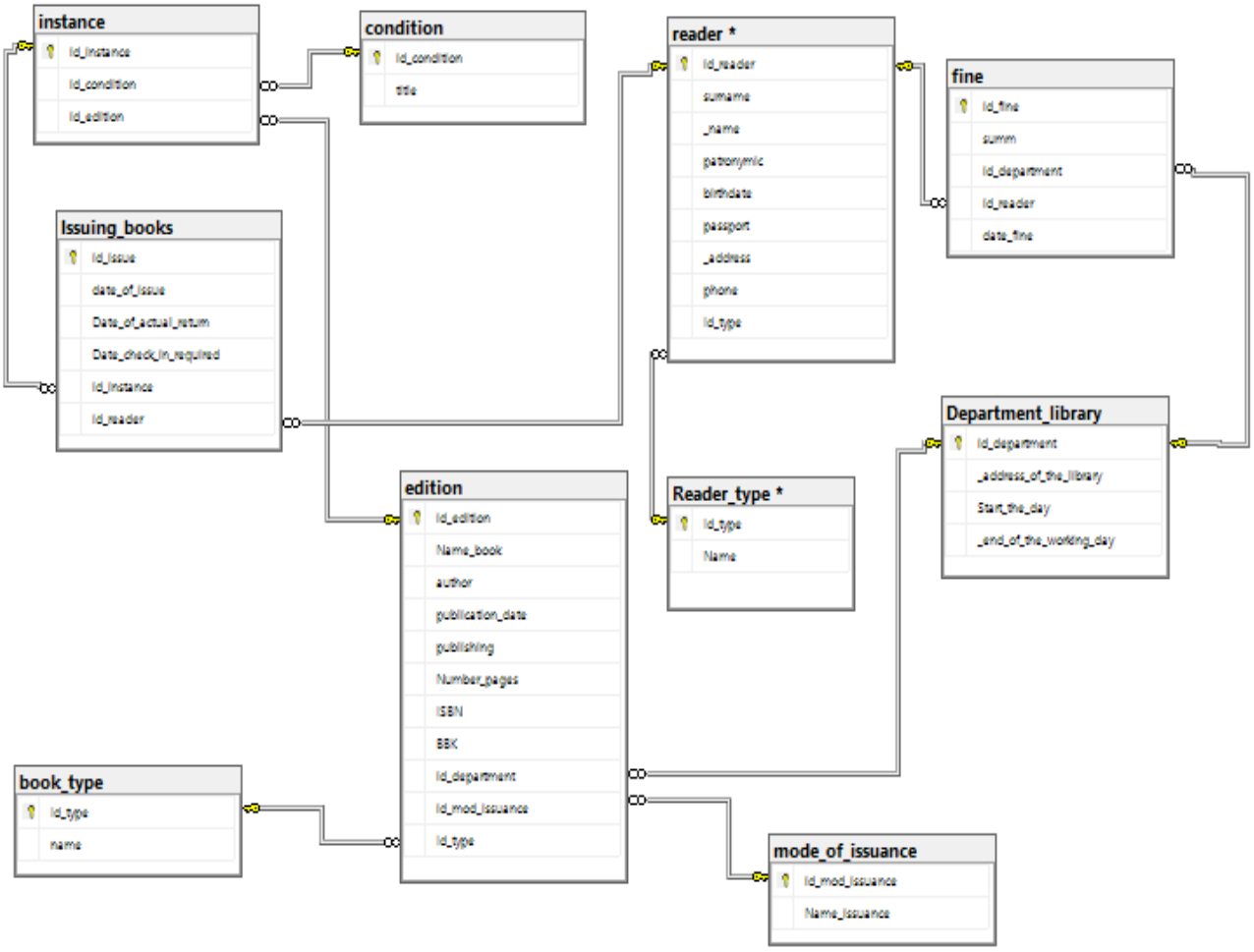


Рисунок 38 – Схема базы данных

Схема базы данных, полученная в СУБД MS SQL Server и физическая модель базы данных полностью совпадают, следовательно, можно сделать вывод, что база данных реализована корректно.

3.2 Реализация программных средств

Так как система состоит из двух подсистем: подсистема Председатель и подсистема кинолог то возможно рассматривать реализацию этих подсистем как самостоятельные компоненты.

Программный код приведен в приложении Б.

3.2.1 Реализация подсистемы «Председатель»

Диаграмма компонентов приложения представлена на рисунке 39.

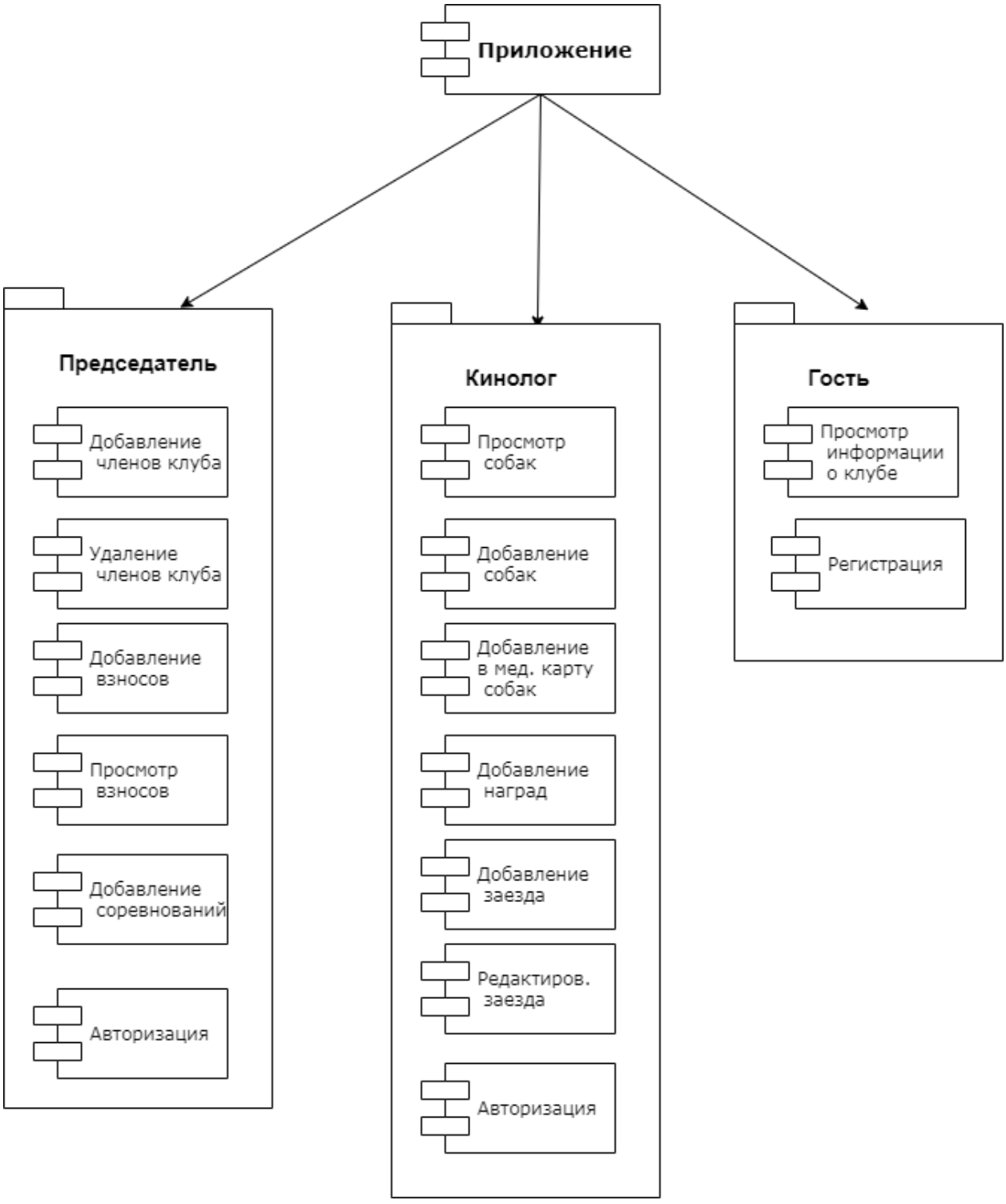


Рисунок 39 – Диаграмма компонентов подсистемы председателя

Описание использованные программных компонентов представлено ниже:

– Admin_panel – исходных код на языке C# запуск приложения;

– Load Data – исходных код на языке C#, содержащий реализацию функций вкладки «Учет владельцев собак»

– LoadData1 – исходных код на языке C#, содержащий описание функций вкладки «Учет уплаты членских взносов»

– LoadData2 – исходных код на языке C#, содержащий описание функций вкладки «Управление соревнованиями»;

– LoadData3 – исходных код на языке C#, содержащий описание функций «Управление аккаунтами»;

– db.OpenConnection – исходных код на языке C#, содержащий описание присоединение к базе данных на сервере;

– dataGridView1.Rows.Add – исходных код на языке C#, содержащий реализацию функций вкладки «Учет уплаты членских взносов»

– dataGridView2.Rows.Add – исходных код на языке C# содержащий реализацию функций вкладки «Управление соревнованиями»;

– dataGridView3.Rows.Add – исходных код на языке C# содержащий реализацию класса функций вкладки «Управление соревнованиями»;

– dataGridView4.Rows.Add – исходных код на языке C# содержащий реализацию функций «Управление аккаунтами»;

– DB.Dbf – файл с базой данных на сервере;

– Panel_kin – исходных код на языке XAML, содержащий описание функций подсистемы «Кинолог»;

– Get.Connection – исходных код на языке C#, содержащий реализацию функции присоединения к базе данных;

– Get.Rase – исходных код на языке C#, содержащий реализацию функций вкладки «Формирование заездов»;

– Get.dog_rewards – исходных код на языке C#, содержащий реализацию основных функций вкладки «Учет наград»;

– Get.dogs – исходных код на языке C#, содержащий реализацию функцмий вкладки «Учет собак»;

– Med_passport_dog – исходных код на языке C#, содержащий реализацию функции формирования медицинских паспортов собак.

3.2.2 Представление развертывания

Одним из способов моделирования статического вида системы с точки зрения развертывания является диаграмма развертывания. Диаграмма развертывания предоставляет физическое расположение системы, показывая на каком физическом оборудовании запускается та или иная составляющая программного обеспечения.

Диаграмма развертывания представлена на рисунке 40.

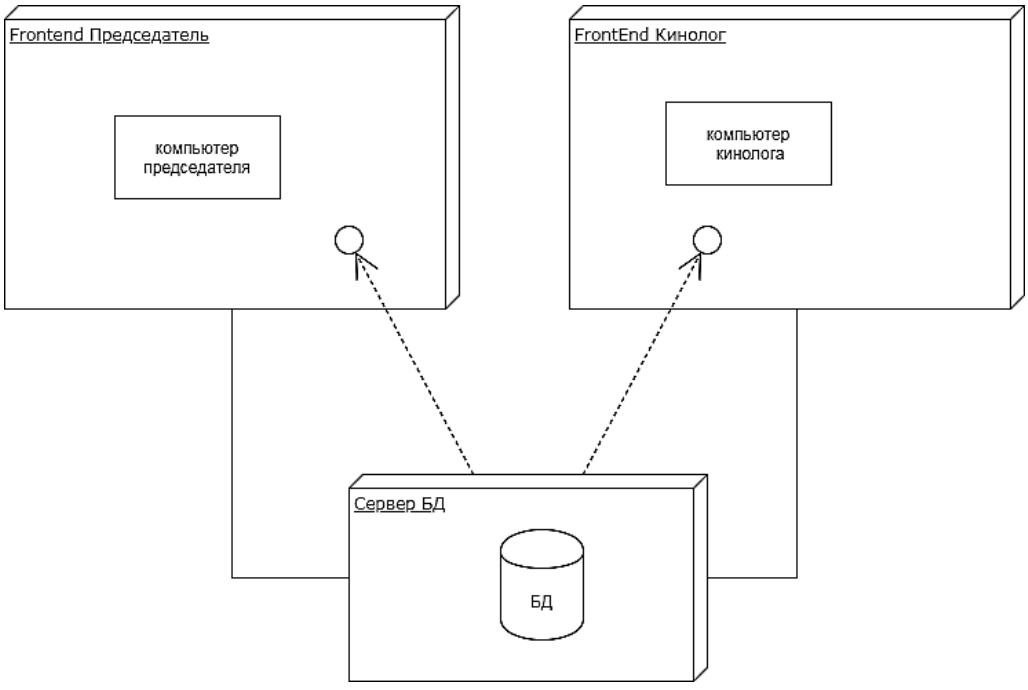


Рисунок 40 – Диаграмма развертывания

Как видно из диаграммы развертывания клиентское приложение разворачивается как на рабочих станциях библиотекаря, так и на рабочих станциях заведующего библиотечным фондом; а база данных разворачивается на отдельном сервере.

4 Тестирование информационной системы клуба федерации ездового спорта

4.1.1 Функциональное тестирование

Функциональное тестирование – процесс, основанный на заранее известном (в некоторых случая предполагаемом) поведении пользователя, основанный в первую очередь на детальном анализе и изучении функциональной спецификации приложения, системы или небольшого модуля (компонента).

Функциональное тестирование является одним из ключевых видов тестирования, задача которого – установить соответствие разработанного программного обеспечения исходным функциональным требованиям заказчика. То есть проведение функционального тестирования позволяет проверить способность информационной системы в определенных условиях решать задачи, нужные пользователям.

Результаты тестирования приведены в приложении Г.

Тестовые наборы приведены в таблице 20.

Таблица 20 – Тестовые наборы для функционального тестирования

Вариант использования	**Тест**	**Ожидаемый результат**	**Результат**
1	2	3	4
Вход в подсистему	Вход осуществляется для председателя Логин: chairman Пароль: ch1234	Открытие подсистема председателя	Тест выполнен, открыто подсистема председателя. (рисунок Г.1)
	Вход осуществляется для кинолога. Логин: dog_hand Пароль: dh1234	Открытие подсистемы кинолога	Тест выполнен, открыто подсистема кинолога (рисунок Г.2)

Продолжение таблицы 20

1	2	3	4
Вход в подсистему	Вход для несуществующего пользователя. Логин: 123456 Пароль: 123456	Вывод сообщения об ошибке.	Тест выполнен, выдано сообщение об ошибке, так как такого пользователя не существует. (рисунок Г.3)
Добавить члена клуба	№ Книжки_владельца:1003 Фамилия: Петров Имя: Иван Отчество: Петрович Год рождения: 1961 Год вступления: 2017:	Добавление пользователя в БД и вывод сообщения об успехе.	Тест выполнен. (рисунок Г.4)
	№ Книжки_владельца:1003 Фамилия: Петров Имя: Иван Отчество: Петрович Год рождения: 3019 Год вступления: 2017:	Вывод сообщения об ошибке.	Тест выполнен, выдано сообщение об ошибке. (рисунок Г.5)
Изменить счета оплаты	Для изменения состояния счета оплаты с не оплачено на оплачено	Состояние счета становится – оплачен, сообщение об успехе операции	Тест выполнен. (рисунок Г.6)
Изменить информацию о собаке	Исходные данные из теста №4 изменяются на следующие: № Книжки_владельца:1111 Фамилия: Иванов Имя: Петр Отчество: Сергеевич Год рождения: 1985 Год вступления: 2019:	Будет выдано сообщение об успехе и изменены данные члена клуба	Тест выполнен, выдано сообщение об ошибке. (рисунок Г.7)
Найти медицинский паспорт собаки	Вводится № медицинского паспорта собаки	В таблице останутся только медицинские данные для животного с таким клеймом	Тест выполнен. (рисунок Г.10)
Найти награды собаки	Вводится №_Клейма собаки	В таблице останутся только награды относящиеся с этому животному.	Тест выполнен. (рисунок Г.11)

Окончание таблицы 20

1	2	3	4
Добавить награду	Выполняется ввод данных №_Клейма собаки 333 ID соревнования 3 ID награды 1 IB титула 4	В таблице наград с собаки с № клейма 333 появится Гранд при Диплом чемпион	Тест выполнен. (рисунок Г.12)
Добавить собаку	№_Книжки владельца:1010 №_Клейма собаки Ф33 ID Породы собаки 1 №_Медицинского паспорта 1066 Кличка Тарри Пол: кобель Год рождения 2016	Сообщение об ошибке.	Тест выполнен. (рисунок Г.13)
Добавить собаку	№_Книжки владельца:1010 №_Клейма собаки 333 ID Породы собаки 1 №_Медицинского паспорта 1066 Кличка Тарри Пол: кобель Год рождения 2016	Добавление собаки в базу	Тест выполнен. (рисунок Г.14)
Удалить собаку	№_Клемма собаки для удаления	Удаление собаки из из списка.	Тест выполнен. (рисунок Г.15)
Найти полную информацию о собаке	Вводится №_Клейма собаки 333.	В таблице результат поиска будет 1010 333 Хаски 1066 Тарри Кобель 2016 Победитель гран при 2019 г Кани-кросс одиночка	Тест выполнен. (рисунок Г.16)

Результаты функционального тестирования показали, что приложение работает корректно и надежно.

4.1.2 Модульное тестирование

Для вариантов использования, где от пользователя требуется заполнение полей напрямую, т.е. он не осуществляет выбор из списка или календаря, было принято решение провести автоматизированное тестирование. Это варианты использования:

– добавление члена клуба;

– добавление счета на оплату;

– изменение информации о члене клуба.

Модульное или unit-тестирование позволяет провести более глубокое исследование и анализ работоспособности отдельных модулей программного обеспечения. Существуют каркасы автономного тестирования, избавляющие пользователя от ручного воспроизведения тестов [13].

Результаты тестирования приведены на рисунке 41.

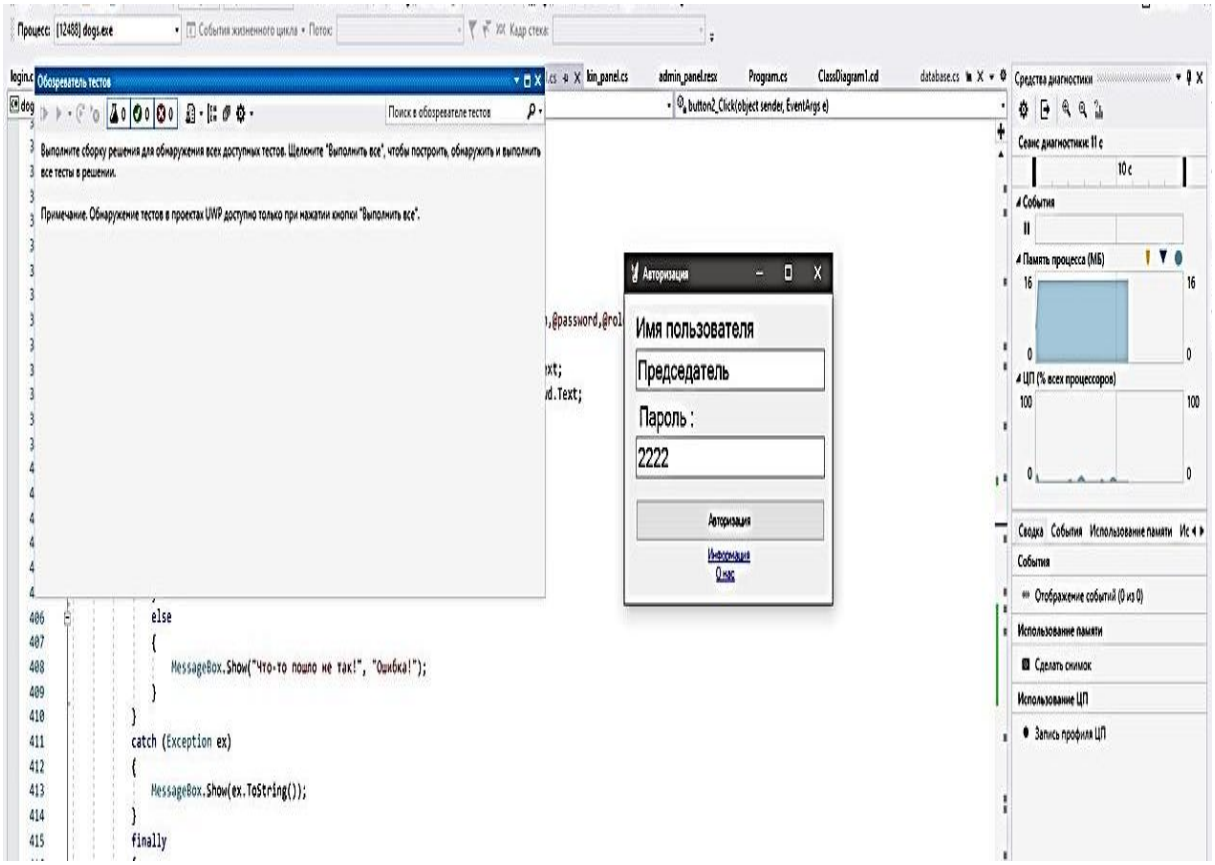


Рисунок 41 – Результаты модульного тестирования

Общее время тестирования модуля составило 2 секунды, также результаты модульного тестирования совпало с результатами функционального тестирования, из чего можно сделать вывод что тестирования было выполнено корректно.

Встроенные средства Visual Studio 2019 позволяют оценить процент покрытия блоков исходного кода тестами и сгенерировать соответствующий отчет.

Из проведенного анализа покрытия следует, что покрытие рассматриваемых классов находится на достаточном уровне, что свидетельствует о надежности исходного кода.

Листинги тестовых проектов приведены в приложении Д.

4.2 Метрики кода

Применение метрик позволяет руководителям проектов и предприятий изучить сложность разработанного или даже разрабатываемого проекта, оценить объем работ, стилистику разрабатываемой программы и усилия, потраченные каждым разработчиком для реализации того или иного решения [13].

Примером часто используемой метрики является комплексный показатель качества кода – Maintainability Index. Рассчитывается метрика по формуле:

$$MI = \max(0,(171 - 5.2 * \ln(HV) - 0.23 * CC - 16.2 * \ln(LoC)) * 100/171),$$

где HV (Halstead Volume) – вычислительная сложность;

CC (Cyclomatic Complexity) – показывает структурную сложность кода, т.е. количество различных ветвей в коде;

LoC (Lines of Code) – количество строк кода.

Комплексный показатель качества кода может принимать значения от 0 до 100 и показывает относительную сложность поддержки кода. Чем больше значение этой метрики, тем легче поддерживать код.

Для определения вычислительной сложности программы сначала следует рассчитать словарь и длину программы, а затем вычислить на их основе HV. Словарь программы рассчитывается по формуле:

$$\eta = \eta_1 + \eta_2,$$

где η_1 – число простых операторов и операций в программе;

η_2 – число простых операндов (констант и переменных) в программе.

Длина программы рассчитывается по формуле:

$$N = N_1 + N_2,$$

где N_1 – общее число простых операторов и операций в программе;

N_2 – общее число простых операндов в программе.

Вычислительная сложность рассчитывается по формуле:

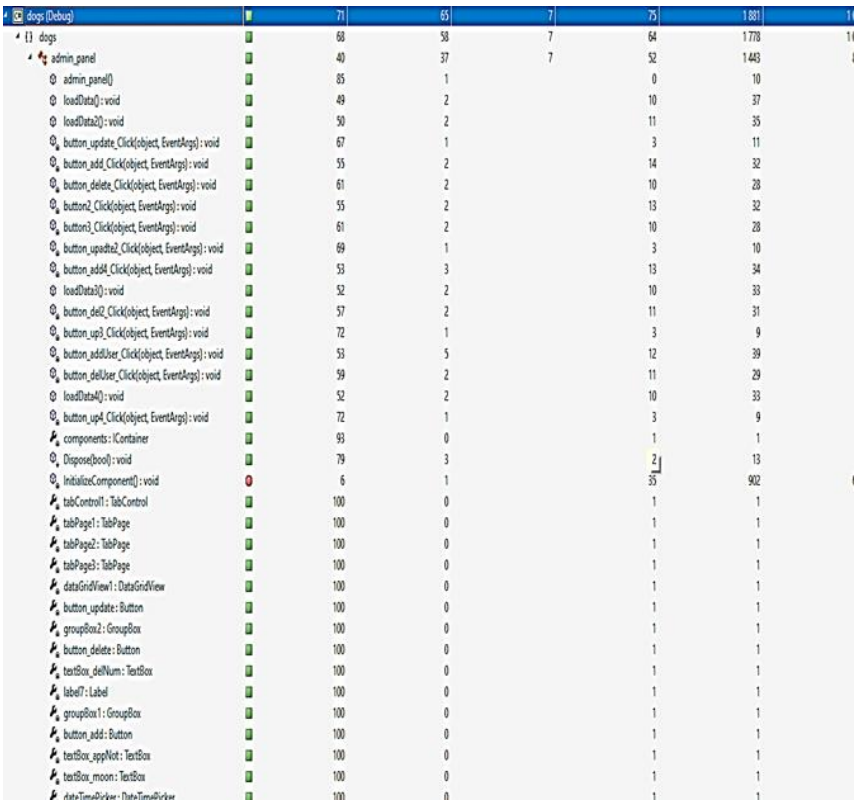
$$\mathrm{H}V = N \log_2 \eta.$$

Структурную сложность программы можно рассчитать по формуле:

CC = (количество операторов if) + (количество операторов цикла) +1.

Если полученное значение CC больше 10, то это свидетельствует о структурной сложности программы, что затрудняет ее функциональное тестирование (методом «белого ящика») и требует упрощения программы [14].

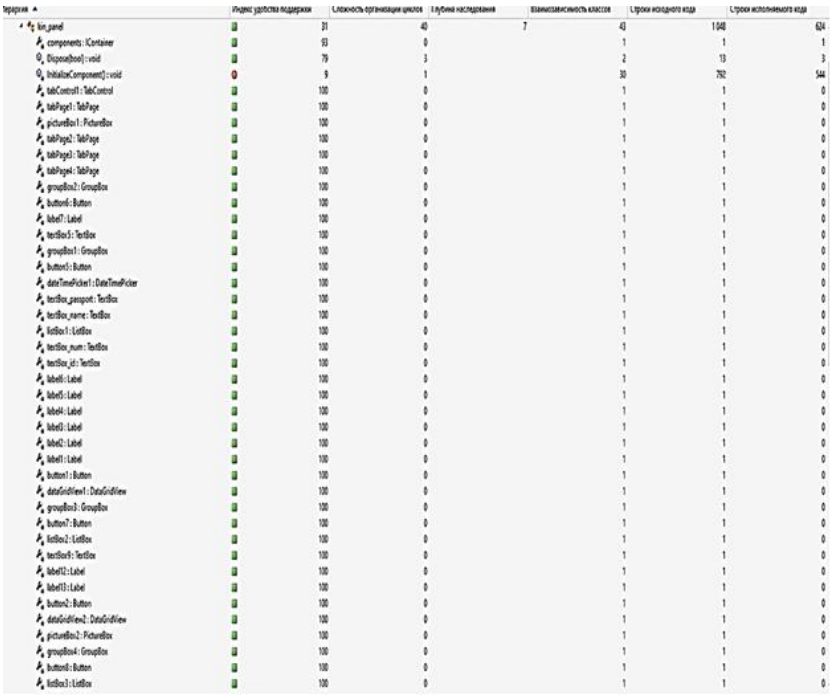
Visual Studio 2019 позволяет вычислять метрики автоматически. Произведем вычисления для разрабатываемого программного обеспечения. Метрики исходного кода подсистемы Председатель, представлены на рисунке 42.



dogs (Debug)	71	65	7	75	1 881	1
{} dogs	68	58	7	64	1 778	1
admin_panel	40	37	7	52	1 443	
admin_panel()	85	1		0	10	
loadData() : void	49	2		10	37	
loadData2() : void	50	2		11	35	
button_update_Click(object, EventArgs) : void	67	1		3	11	
button_add_Click(object, EventArgs) : void	55	2		14	32	
button_delete_Click(object, EventArgs) : void	61	2		10	28	
button2_Click(object, EventArgs) : void	55	2		13	32	
button3_Click(object, EventArgs) : void	61	2		10	28	
button_upadte2_Click(object, EventArgs) : void	69	1		3	10	
button_add4_Click(object, EventArgs) : void	53	3		13	34	
loadData3() : void	52	2		10	33	
button_del2_Click(object, EventArgs) : void	57	2		11	31	
button_up3_Click(object, EventArgs) : void	72	1		3	9	
button_addUser_Click(object, EventArgs) : void	53	5		12	39	
button_delUser_Click(object, EventArgs) : void	59	2		11	29	
loadData4() : void	52	2		10	33	
button_up4_Click(object, EventArgs) : void	72	1		3	9	
components : IContainer	93	0		1	1	
Dispose(bool) : void	79	3		2	13	
InitializeComponent() : void	6	1		35	902	
tabControl1 : TabControl	100	0		1	1	
tabPage1 : TabPage	100	0		1	1	
tabPage2 : TabPage	100	0		1	1	
tabPage3 : TabPage	100	0		1	1	
dataGridView1 : DataGridView	100	0		1	1	
button_update : Button	100	0		1	1	
groupBox2 : GroupBox	100	0		1	1	
button_delete : Button	100	0		1	1	
textBox_delNum : TextBox	100	0		1	1	
label7 : Label	100	0		1	1	
groupBox1 : GroupBox	100	0		1	1	
button_add : Button	100	0		1	1	
textBox_appNot : TextBox	100	0		1	1	
textBox_moon : TextBox	100	0		1	1	

Рисунок 42 – Метрики исходного кода подсистемы председатель

Метрики исходного кода подсистемы Кинолог фондом представлены на рисунке 43.



Иерархия	Индекс удобства поддержки	Сложность организации циклов	Глубина наследования	Взаимозависимость классов	Строки исходного кода	Строки исполняемого кода
kin_panel	31	40	7	43	1 048	624
components : IContainer	93	0		1	1	1
Dispose(bool) : void	79	3		2	13	3
InitializeComponent() : void	9	1		30	792	544
tabControl1 : TabControl	100	0		1	1	0
tabPage1 : TabPage	100	0		1	1	0
pictureBox1 : PictureBox	100	0		1	1	0
tabPage2 : TabPage	100	0		1	1	0
tabPage3 : TabPage	100	0		1	1	0
tabPage4 : TabPage	100	0		1	1	0
groupBox2 : GroupBox	100	0		1	1	0
button6 : Button	100	0		1	1	0
label7 : Label	100	0		1	1	0
textBox3 : TextBox	100	0		1	1	0
groupBox1 : GroupBox	100	0		1	1	0
button5 : Button	100	0		1	1	0
dateTimePicker1 : DateTimePicker	100	0		1	1	0
textBox_passport : TextBox	100	0		1	1	0
textBox_name : TextBox	100	0		1	1	0
listBox1 : ListBox	100	0		1	1	0
textBox_num : TextBox	100	0		1	1	0
textBox_id : TextBox	100	0		1	1	0
label6 : Label	100	0		1	1	0
label5 : Label	100	0		1	1	0
label4 : Label	100	0		1	1	0
label3 : Label	100	0		1	1	0
label2 : Label	100	0		1	1	0
label1 : Label	100	0		1	1	0
button1 : Button	100	0		1	1	0
dataGridView1 : DataGridView	100	0		1	1	0
groupBox3 : GroupBox	100	0		1	1	0
button7 : Button	100	0		1	1	0
listBox2 : ListBox	100	0		1	1	0
textBox9 : TextBox	100	0		1	1	0
label12 : Label	100	0		1	1	0
label13 : Label	100	0		1	1	0
button2 : Button	100	0		1	1	0
dataGridView2 : DataGridView	100	0		1	1	0
pictureBox2 : PictureBox	100	0		1	1	0
groupBox4 : GroupBox	100	0		1	1	0
button8 : Button	100	0		1	1	0
listBox3 : ListBox	100	0		1	1	0

Рисунок 43 – Метрики исходного кода подсистемы Кинолог

Maintainability Index – комплексный показатель качества кода. Этот показатель разработан специалистами из Carnegie Mellon Software Engineering Institutex[16].

Рассчитывается метрика по следующей формуле:
MI = MAX(0, (171 — 5.2 * ln(HV) — 0.23 * CC — 16.2 * ln(LoC)) * 100 / 171)
В данном проекте комплексный показатель кода (MI) находится на высоком уровне. Средняя отметка комплексного показателя составляет97. Это свидетельствует об относительной простоте поддержки исходного кода.

Структурная сложность кода (CC) находится на среднем уровне.

В данном проекте большой глубиной наследования обладают классы-окна, описывающие графический интерфейс приложения, остальные классы обладают глубиной наследования не более 2. Это говорит о том, что человеку, не знакомому с данным проектом, будет легче понять, где определяются или переопределяются конкретные методы и поля класса.

Чем больше показатель связанности (объединения) классов, тем сложнее поддерживать данные классы. Как видно из результатов, показатель связанности относительно невысокий.

Таким образом, тестирование показало соответствие работы программы предъявленным функциональным требованием, что обосновано результатами Unit test, метриками программного кода и представленными тестовыми наборами данных.

5 Планирование проекта и оценка бюджета

Одним из основополагающих видов планирования проектов является структурное планирование, которое включает в себя следующие этапы:

– разбиение проекта на совокупность отдельных работ, выполнение которых необходимо для реализации проекта;

– построение сетевого графика, описывающего последовательность выполнения работ;

– оценка временных характеристик работ и анализ сетевого графика.

Управление проектами – область деятельности, в ходе которой определяются и достигаются цели проекта при балансировании между объемом работ, материальными и трудовыми ресурсами, качеством, временем, и рисками. Ключевым фактором успеха проектного управления является наличие четкого, заранее определенного плана [15].

Процесс планирования в среде Microsoft Project имеет свою специфику, которая состоит в следующей последовательности действий:

– создание плана проекта на основе анализа предметной области и функциональных требований;

– определение и структурирование списка задач проекта, определение длительности каждой задачи и зависимости между ними;

– создание списка ресурсов;

– назначение ресурсов по задачам.

На основе введенной информации Microsoft Project создает расписание, которое затем можно анализировать, настраивать и оптимизировать.

В ходе планирования проекта было выделено 23 задачи. Перечень задач проекта, их виды, связи, длительность и предшественники приведены в таблице 20.

Таблица 21 – Задачи проекта

Название	Вид Задачи	Предшественники	Длительность
1	2	3	4
Начало реализации проекта	Веха		-
Анализ предметной области	Фаза		-
Анализ программных аналогов	Задача	1	1
Постановка задачи	Задача	3	1
Анализ предметной области завершен	Веха	4	-
Анализ бизнес-процессов предметной области	Задача	5	1
Анализ требований	Фаза	6	-
Анализ функциональных требований	Задача	7	1
Анализ нефункциональных требований	Задача	6	1
Анализ требований завершен	Веха	9	-
Выбор инструментальных средств	Задача	7	1
Выбор системных программных средств	Задача	11	1
Проектирование	Фаза	10	-
Проектирование структур данных	Задача	11	1
Проектирование базы данных	Задача	11	3
Проектирование программных средств подсистемы председателя	Задача	14	3
Проектирование программных средств подсистемы кинолога	Задача	15	3
Проектирование пользовательского подсистемы председателя	Задача	16	2
Проектирование пользовательского интерфейса приложения подсистемы кинолога	Задача	17	2
Проектирование завершено	Веха	18;19	-
Реализация	Фаза	20	-

Продолжение таблицы 21

1	2	3	4
Реализация программных средств базы данных	Задача	20	2
Реализация программных подсистемы председателя	Задача	22	3
Реализация программных средств подсистемы кинолога	Задача	23	3
Реализация завершена	Веха	24	-
Отладка и тестирование	Фаза	21	-
Отладка серверной части	Задача	25	1
Отладка клиентской части	Задача	27	2
Автономное тестирование	Задача	28	1
Рефакторинг	Задача	29	3
Анализ качества кода	Задача	30	3
Отладка и тестирование завершены	Веха	31	-
Завершение проекта	Фаза	32	-
Составление программной документации	Задача	33	2
Сдача в эксплуатацию	Задача	34	1
Проект завершен	Веха	35	-

Дата начала проекта 01.02.20. Необходимо внести выделенные задачи в созданный проект в среде Microsoft Project, а также установить их длительность и предшественников, после чего будет построена диаграмма Ганта, отображающую следующие параметры проекта:

– структуру работ, полученную на основе сетевого графика;

– состав используемых ресурсов и их распределение между работами;

– календарные даты, к которым привязываются моменты начала и завершения работ [16].

После определения состава работ и связей между ними необходимо определить состав ресурсов для выполнения задач проекта без которых полноты картины анализа просто не достичь, ведь затраты являются ключевой составной, которая определяет возможности и границы проекта, а также его рентабельность.

Ресурсы и затраты приведены в таблице 22.

Таблица 22 – Ресурсы и затраты проекта

Название ресурса	**Тип**	**Затраты**			**Затраты на исп.**
		Таблица норм	**Станд. ставка**	**Ставка сверхур.**	
1	2	3	4	5	6
Программист 1 разработчик БД	Т	А	45000 р./мес	500р./ч	-
Программист 2 разработчик клиентской части	Т	А	60000 р./мес	500р./ч	-
Тестировщик (тестменеджер)	Т	А	35000 р./мес	400р./ч	-
Компьютер (ноутбук)	М	А	85000 руб.		-
Лицензионное программное обеспечение (подписка)	М	А	72000 руб.		
Принтер	М	А	5300 руб.		-
Бумага	М	А	400 руб.		-
Платный Git хостинг	М	А	5000 руб.		-
Интернет	З				-
Мобильная связь	З				-

Распределение работ проекта по задачам приведено в таблице 23.

Таблица 23 – Распределение работ проекта по задачам

Название	**Ресурс**	**Единицы (затраты)**	**Таблица норм затрат**
1	2	3	4
Начало реализации проекта	Бумага Компьютер Принтер	1 2 1	A A A
Анализ программных аналогов	Интернет Программист 1 Программист 2	250 100 150	A A
Постановка задачи	Программист 2	100	A
Анализ функциональных требований	Программист 2	100	A
Анализ нефункциональных требований	Программист 2	100	A
Выбор инструментальных средств	Интернет Программист 1 Программист 2	250 100	A A
Выбор системных программных средств	Интернет Программист 1 Программист 2	250 100	A A
Проектирование структур данных	Программист 1	100	A
Проектирование базы данных	Программист 1	100	A
Проектирование программных средств подсистемы председателя	Программист 2	50	A
Проектирование программных средств подсистемы кинолога	Программист 2	50	A
Проектирование пользовательского интерфейса приложения библиотекаря	Программист 2	50	A
Проектирование пользовательского интерфейса подсистемы председателя	Программист 2	50	A

Продолжение таблицы 23

1	2	3	4
Реализация программных средств базы данных	Программист 2	100	А
Реализация программных средств подсистемы кинолога	Программист 2	50	А
Отладка БД	Программист 1 Тестировщик	50 50	А А
Отладка оконного приложения	Программист 2 Тестировщик	50 50	А А
Автономное тестирование	Тестировщик	100	А
Рефакторинг	Программист 2 Git хостинг Интернет	100	А
Анализ качества кода	Программист 1 Git хостинг Интернет	100	А
Составление программной документации	Программист 1 Программист 2 Интернет Git хостинг	100 1 1 200	А А А А
Сдача в эксплуатацию	Программист 2	100	А

Результат внесения данных в проект приведен на рисунке 44.

	Название ресурса	Тип	Единицы измерения материалов	Краткое название	Группа	Макс. единиц	Стандартная ставка	Ставка сверхурочных	Затраты на использ.	Начисление	Базовый календарь
1	Программист 1	Трудовой		П		100%	20 000,00 ₽/мес	200,00 ₽/ч	0,00 ₽	Пропорциональное	Стандартный
2	Тестировщик	Трудовой		Т		100%	15 000,00 ₽/мес	100,00 ₽/ч	0,00 ₽	Пропорциональное	Стандартный
3	Компьютер	Материальный		К			15 000,00 ₽		0,00 ₽	Пропорциональное	
4	Принтер	Материальный		Пр			3 000,00 ₽		0,00 ₽	Пропорциональное	
5	Бумага	Материальный		Б			200,00 ₽		0,00 ₽	Пропорциональное	
6	Флэш-накопитель	Материальный		Ф			500,00 ₽		0,00 ₽	Пропорциональное	
7	Интернет	Затраты		И						Пропорциональное	
8	Мобильная связь	Затраты		М						Пропорциональное	

Рисунок 44 – Лист ресурсов

После внесения в проект всех необходимых данных среда Microsoft Project построит диаграмму Ганта с указанием исполнителей работ, их стоимостью, процентом загрузки исполнителя, датами начала и завершения проекта, а также вехами проекта.

Полученная диаграмма Ганта для разрабатываемого проекта приведена на рисунке 45.

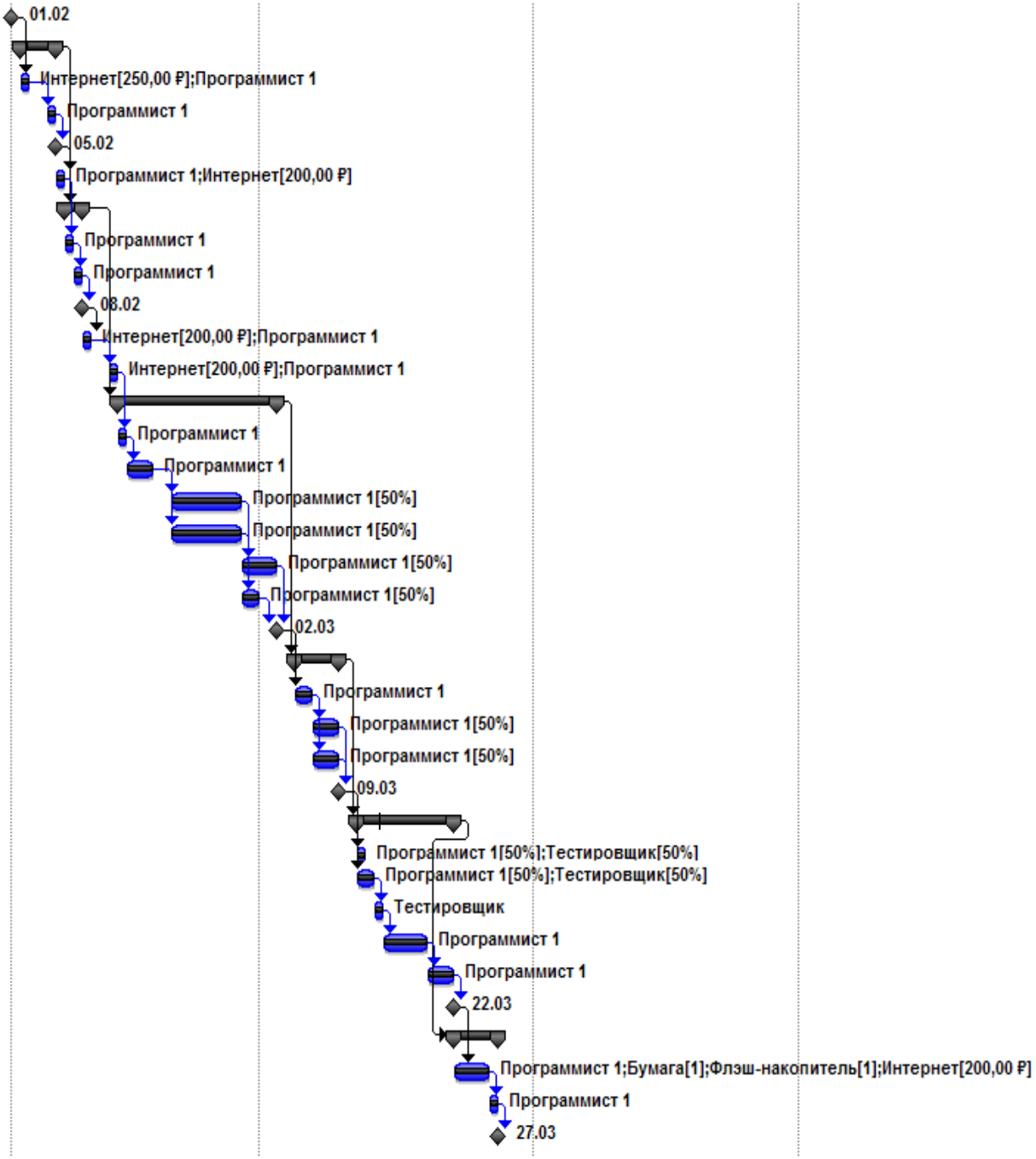


Рисунок 45 – Диаграмма Ганта

Сводка по проекту представлена на рисунке 46.

Даты			
Начало:	Чт 01.02.18	Окончание:	Вт 27.03.18
Баз. начало:	НД	Баз. окончание:	НД
Факт. начало:	Чт 01.02.18	Факт. окончание:	Вт 27.03.18
Отклон. начала:	0 дней	Отклон. окончания:	0 дней

Длительность			
Текущий план:	39 дней	Оставшаяся:	0 дней
Базовый план:	0 дней?	Фактическая:	39 дней
Отклонение:	39 дней	Процент завершения	100%

Трудозатраты			
Текущий план:	304 ч	Оставшиеся:	0 ч
Базовый план:	0 ч	Фактические:	304 ч
Отклонение:	304 ч	Процент завершения	100%

Затраты			
Текущий план:	72 325,00 ₽	Оставшиеся:	1 050,00 ₽
Базовый план:	0,00 ₽	Фактические:	71 275,00 ₽
Отклонение:	72 325,00 ₽		

Состояние задачи		Состояние ресурса	
Еще не начатые задачи	0	Трудовые ресурсы:	4
Выполняющиеся задачи	0	Трудовые ресурсы с прев. дост	0
Завершенные задачи:	36	Материальные ресурсы:	4
Итого задач:	36	Итого ресурсов:	8

Рисунок 46 – Сводка по проекту

На полученной сводке указана дата начала и окончания работ, длительность в днях, процент выполнения на данном этапе, а также процент завершения по каждой из задач верхнего уровня. В данном случае все задачи завершены успешно, выполнены на 100 %, как и планировалось.

Из полученного отчета можно сделать вывод, что на реализацию проекта потребовалось 39 рабочих дней, а бюджет проекта составил 72 325 рублей.

Заключение

При выполнении выпускной квалификационной работы было разработано программное обеспечение для информационной системы Федерации ездового спорта.

Клиентское приложение реализовано с помощью языка C# в интегрированной среде разработки Microsoft Visual Studio.

Серверное приложение реализовано с помощью языка SQL в интегрированной среде разработки Microsoft SQL Server.

На этапе анализа качества кода комплексный показатель находится на достаточно высоком уровне и лишь в некоторых классах опускается ниже 80%, что свидетельствует об относительной простоте поддержки исходного кода.

Также было выполнено планирование проекта с использованием программного продукта Microsoft Project. Длительность выпылнения проекта равна 39 дням, а бюджет равен 72 325 рублей. Оценка сроков и бюджета говорит о том, что данная разработка экономически выгодна.

В ходе работы было проведено функциональное тестирование. На основе анализа результатов тестирования можно сделать вывод, что разработанное программное обеспечение работает корректно.

Таким образом, в результате выполнения выпускной квалификационной работы было разработано требуемое программное обеспечение отвечающее всем требованиям технического задания.

Все задачи поставленные в выпускной квалификационной работе выполнены цель достигнута.

Список использованных источников

1. Севрюкова А. А. Автоматизированные библиотечные системы // Молодой ученый. – 2015. – №13. – С. 857-859 [Электронный ресурс]. – URL: https://moluch.ru/archive/93/20599/ (дата обращения: 01.05.2018).

2. О выборе автоматизированной информационной библиотечной системы для библиотеки ИПМ / М.М.Горбунов-Посадов [и др.] // Препринты ИПМ им. М.В.Келдыша. – 2011. – №2. – 32с [Электронный ресурс]. – URL: http://library.keldysh.ru/preprint.asp?id=2011-2 (дата обращения: 01.05.2018).

3. Презентация на тему: Учебный курс для школьных библиотекарей по программе MARK-SQL АРМ Каталогизация MARK-SQL // MyShared [Электронный ресурс]. – URL: http://www.myshared.ru/slide/7303/ (дата обращения: 01.05.2018).

4. АБИС семейства МАРК-SQL // Информ-система – софт для библиотек [Электронный ресурс]. – URL: http://www.informsystema.ru/ (дата обращения: 01.05.2018).

5. Absotheque UNICODE // LIBER [Электронный ресурс]. – URL: https://www.libermedia.ru/catalog/index.php?ELEMENT_ID=1 (дата обращения: 01.05.2018).

6. Продукты семейства ИРБИС// Ассоциация ЭБНИТ [Электронный ресурс]. – URL: http://www.elnit.org (дата обращения: 01.05.2018).

7. Универсальный инструментарий автоматизации библиотек и библиотечных сетей // ДИТ-М | OPAC-Global [Электронный ресурс]. – URL: http://www.ditm.ru/opac-global (дата обращения: 01.05.2018).

8. Арлоу, Д. UML 2 и Унифицированный процесс. Второе издание/Д. Арлоу, А. Нейштадт. – СПб.: Питер, 2006. – 624 с.

9. Макконнелл, С. Совершенный код. Мастер-класс. / С. Макконнелл. – М.: Русская редакция, 2013. – 893 с.

10. Статические показатели и основы поведения статического исследования // Studbooks.net [Электронный ресурс]. – URL: http://studbooks.net/522117/kulturologiya/statisticheskie_pokazateli_osnovy_provedeniya_statisticheskogo_issledovaniya (дата обращения: 05.05.2018).

11. Регистрационная карточка читателя // Муниципальное казенное учреждение культуры Межпоселенческая библиотека Черемховского района [Электронный ресурс]. – URL: http://lib-cher-rayon.ru/jirbis2/images/docs/V_pomosh/Registr%20kartocka%20zitately.pdf (дата обращения: 05.05.2018).

12. Адаптер (Adapter) // METANIT.COM Сайт о программировании [Электронный ресурс]. – URL: https://metanit.com/sharp/patterns/4.2.php (дата обращения: 05.05.2018).

13. Тестирование программного обеспечнения. Фундаментальные концепции менеджмент бизнес-приложений: Пер. с англ./ Сэм, Джек Фолк, Енг Кек Нгуен. – К.: Издательство «ДиаСофт», 2001. – 544 с.

14. Выполнение выпускной квалификационной работы. Учебное пособие для бакалавров по направлениям 09.03.04 Программная инженерия, 09.03.02 «Информационные системы и технологии» / под ред. П. П. Макарычева. – Пенза: Изд-во ПГУ, 2015. – 150 с.

15. Куперштейн, В. И. Microsoft Project 2010 в управлении проектами. / В. И. Куперштейн, А.В. Цветкова. – СПб.: БХВ-Петербург, 2011. – 416 с.

16. Шкрыль, А.А. MS Project 2007: современное управление проектами. / А.А. Шкрыль. – СПб.: БХВ-Петербург, 2007. – 256 с.

Приложение А
ГЛОССАРИЙ
(справочное)

Автоматизированная информационная система (АИС) – это комплекс программных, технических, информационных, организационно-технологических средств, предназначенный для решения задач справочно-информационного обслуживания или информационного обеспечения пользователей.

Автоматизированная библиотечная информационная система (АБИС) – система планирования ресурсов предприятий для библиотеки, используемая для отслеживания библиотечных фондов, от их заказа и приобретения до выдачи посетителям библиотек.

База данных – представленная в объективной форме совокупность самостоятельных материалов (статей, расчетов, нормативных актов, судебных решений и иных подобных материалов), систематизированных таким образом, чтобы эти материалы могли быть найдены и обработаны с помощью электронной вычислительной машины (ЭВМ).

Объектно-ориентированное программирование (ООП) – это методология программирования, основанная на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определенного класса, а классы образуют иерархию наследования.

Программное обеспечение(ПО) – совокупность программ обработки информации и программных документов, необходимых для эксплуатации этих программ.

Система управления базами данных (СУБД) – совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных.

Приложение Б
ЛИСТИНГ ПРОГРАММЫ
(рекомендуемое)

Листинг Б.1 – Подсистема «Председатель» на языке C#(файл panel_admin.cs)

```
using Microsoft.VisualStudio.TestTools.UnitTesting;
using MySql.Data.MySqlClient;
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 dogs
{
    public partial class admin_panel : Form
    {


            public admin_panel()
            {
                  InitializeComponent();
                  loadData();
                  loadData2();
                  loadData3();
                  loadData4();
            }

            public void loadData()
            {
                  database db = new database();
                  string string_command = "SELECT * FROM `payment`;";
                  MySqlCommand mySqlCommand = new
MySqlCommand(string_command, db.GetConnection());
                  try
                  {
                        db.OpenConnection();
                        MySqlDataReader mySqlDataReader =
mySqlCommand.ExecuteReader();
                        dataGridView1.Columns.Add("ID_Book", "Номер
членской книжки");
                        dataGridView1.Columns.Add("Account_number",
"Номер счета");
                        dataGridView1.Columns.Add("Data_p", "Дата
оплаты");
                        dataGridView1.Columns.Add("Moon", "Месяц");
                        dataGridView1.Columns.Add("Summa", "Сумма");
                        dataGridView1.Columns.Add("App_Not", "Статус
оплаты");
                        dataGridView1.Columns["ID_Book"].Width = 100;
                        dataGridView1.Columns["ID_Book"].ReadOnly = true;
                        dataGridView1.Columns["Account_number"].ReadOnly
= true;
                        dataGridView1.Columns["Data_p"].ReadOnly = true;
                        dataGridView1.Columns["Moon"].ReadOnly = true;
```

```
                    dataGridView1.Columns["Summa"].ReadOnly = true;
                    dataGridView1.Columns["App_Not"].ReadOnly = true;
                    while (mySqlDataReader.Read())
                    {

     dataGridView1.Rows.Add(mySqlDataReader["ID_Book"].ToString(),
mySqlDataReader["Account_number"].ToString(),
mySqlDataReader["Data_p"].ToString(),
mySqlDataReader["Moon"].ToString(),
mySqlDataReader["Summa"].ToString(),
mySqlDataReader["App_Not"].ToString());
                    }
               }
               catch
               {
                    MessageBox.Show("Нет подключения к базе данных,
обратитесь к системному администратору.", "Ошибка!");
               }
               finally
               {
                    db.CloseConnection();
               }
          }

          public void loadData2()
          {
               database db = new database();
               string string_command = "SELECT * FROM `owners`;";
               MySqlCommand mySqlCommand = new
MySqlCommand(string_command, db.GetConnection());
               try
               {
                    db.OpenConnection();
                    MySqlDataReader mySqlDataReader =
mySqlCommand.ExecuteReader();
                    dataGridView2.Columns.Add("ID_Book", "Номер
членской книжки");
                    dataGridView2.Columns.Add("FIO", "ФИО
пользователя");
                    dataGridView2.Columns.Add("Address", "Адресс");
                    dataGridView2.Columns.Add("Year_birth", "Дата
рождения");
                    dataGridView2.Columns.Add("Year_entry", "Дата
вступления");
                    dataGridView2.Columns["ID_Book"].Width = 100;
                    dataGridView2.Columns["ID_Book"].ReadOnly = true;
                    dataGridView2.Columns["FIO"].ReadOnly = true;
                    dataGridView2.Columns["Address"].ReadOnly = true;
                    dataGridView2.Columns["Year_birth"].ReadOnly =
true;
                    dataGridView2.Columns["Year_entry"].ReadOnly =
true;
                    while (mySqlDataReader.Read())
                    {

     dataGridView2.Rows.Add(mySqlDataReader["ID_Book"].ToString(),
mySqlDataReader["FIO"].ToString(),
```

```
mySqlDataReader["Address"].ToString(),
mySqlDataReader["Year_birth"].ToString(),
mySqlDataReader["Year_entry"].ToString());
                    }
                }
                catch (Exception ex)
                {
                    MessageBox.Show(ex.ToString());
                }
                finally
                {
                    db.CloseConnection();
                }
            }

            private void button_update_Click(object sender, EventArgs e)
            {
                dataGridView1.Columns.Remove("ID_Book");
                dataGridView1.Columns.Remove("Account_number");
                dataGridView1.Columns.Remove("Data_p");
                dataGridView1.Columns.Remove("Moon");
                dataGridView1.Columns.Remove("Summa");
                dataGridView1.Columns.Remove("App_Not");
                loadData();
            }

            private void button_add_Click(object sender, EventArgs e)
            {
                database db = new database();
                string command_string = "INSERT INTO `payment`
(Account_number, Data_p, Moon, Summa, App_Not) VALUES
(@Num,@Data,@Moon,@Summ,@App_Not)";
                MySqlCommand mySqlCommand = new
MySqlCommand(command_string, db.GetConnection());
                mySqlCommand.Parameters.Add("@Num",
MySqlDbType.Int32).Value = textBox_num_acc.Text;
                mySqlCommand.Parameters.Add("@Data",
MySqlDbType.DateTime).Value = dateTimePicker.Value;
                mySqlCommand.Parameters.Add("@Moon",
MySqlDbType.VarChar).Value = textBox_moon.Text;
                mySqlCommand.Parameters.Add("@Summ",
MySqlDbType.Int32).Value = textBox_summ.Text;
                mySqlCommand.Parameters.Add("@App_Not",
MySqlDbType.Binary).Value = textBox_appNot.Text;
                try
                {
                    db.OpenConnection();
                    if (mySqlCommand.ExecuteNonQuery() == 1)
                    {
                        MessageBox.Show("Оплата успешно добавлена",
"Успешно");
                    }
                    else
                    {
                        MessageBox.Show("Что-то пошло не так!",
"Ошибка!");
                    }
```

```
                }
                catch (Exception ex)
                {
                    MessageBox.Show(ex.ToString());
                }
                finally
                {
                    db.CloseConnection();
                }
            }

            private void button_delete_Click(object sender, EventArgs e)
            {
                database db = new database();
                string command_string = "DELETE FROM `payment` WHERE
Account_number=@Num";
                MySqlCommand sqlCommand = new
MySqlCommand(command_string, db.GetConnection());
                sqlCommand.Parameters.Add("@Num",
MySqlDbType.Int32).Value = textBox_delNum.Text;
                try
                {
                    db.OpenConnection();
                    if (sqlCommand.ExecuteNonQuery() >= 1)
                    {
                        MessageBox.Show("Оплата успешно удалена!",
"Успешно!");
                    }
                    else
                    {
                        MessageBox.Show("Предмета с таким артикулом
не существует.", "Ошибка!");
                    }
                }
                catch
                {
                    MessageBox.Show("Нет подключения к базе данных.
Обратитесь к администратору системы.", "Ошибка!");
                }
                finally
                {
                    db.CloseConnection();
                }
            }

            private void button2_Click(object sender, EventArgs e)
            {
                unit_test();
                database db = new database();
                string command_string = "INSERT INTO `owners`
(ID_Book, FIO, Address , Year_birth, Year_entry) VALUES (@id,
@fio,@address,@year_birth,@year_empty)";
                MySqlCommand mySqlCommand = new
MySqlCommand(command_string, db.GetConnection());
                mySqlCommand.Parameters.Add("@id",
MySqlDbType.Int32).Value = textBox_id.Text;
```

```
                mySqlCommand.Parameters.Add("@fio",
MySqlDbType.VarChar).Value = textBox_FIO.Text;
                mySqlCommand.Parameters.Add("@address",
MySqlDbType.VarChar).Value = textBox_address.Text;
                mySqlCommand.Parameters.Add("@year_birth",
MySqlDbType.Date).Value = dateTimePicker_year.Value;
                mySqlCommand.Parameters.Add("@year_empty",
MySqlDbType.Date).Value = dateTimePicker_empty.Value;
                try
                {
                      db.OpenConnection();
                      if (mySqlCommand.ExecuteNonQuery() == 1)
                      {
                            MessageBox.Show("Владелец успешно добавлен",
"Успешно");
                      }
                      else
                      {
                            MessageBox.Show("Что-то пошло не так!",
"Ошибка!");
                      }
                }
                catch (Exception ex)
                {
                      MessageBox.Show(ex.ToString());
                }
                finally
                {
                      db.CloseConnection();
                }
          }

          private void button3_Click(object sender, EventArgs e)
          {
                database db = new database();
                string command_string = "DELETE FROM `owners` WHERE
ID_Book=@id_book";
                MySqlCommand sqlCommand = new
MySqlCommand(command_string, db.GetConnection());
                sqlCommand.Parameters.Add("@id_book",
MySqlDbType.Int32).Value = textBox_IDBook.Text;
                try
                {
                      db.OpenConnection();
                      if (sqlCommand.ExecuteNonQuery() >= 1)
                      {
                            MessageBox.Show("Предмет успешно удален!",
"Успешно!");
                      }
                      else
                      {
                            MessageBox.Show("Предмета с таким артикулом
не существует.", "Ошибка!");
                      }
                }
                catch
                {
```

```
                    MessageBox.Show("Нет подключения к базе данных.
Обратитесь к администратору системы.", "Ошибка!");
                }
                finally
                {
                    db.CloseConnection();
                }
            }

            private void button_upadte2_Click(object sender, EventArgs
e)
            {
                dataGridView2.Columns.Remove("ID_Book");
                dataGridView2.Columns.Remove("FIO");
                dataGridView2.Columns.Remove("Address");
                dataGridView2.Columns.Remove("Year_birth");
                dataGridView2.Columns.Remove("Year_entry");
                loadData2();
            }

            [TestMethod]
            public void unit_test()
        {
                int id = 3;
                textBox_id.Text = id.ToString(); ;
                string FIO = "Иванов Иван Иванович";
                textBox_FIO.Text = FIO;
                string address = "12";
                textBox_address.Text = address;
                dateTimePicker_year.Value = new DateTime(2009, 3, 23);
                dateTimePicker_empty.Value = new DateTime(2019, 3,
23);

            }
            private void button_add4_Click(object sender, EventArgs e)
            {
                database db = new database();
                int id = 0;

                if (listBox2.SelectedItem == "Скай джоринг спринт")
                    id = 1;
                if (listBox2.SelectedItem == "Скай джоринг
полумарафон")
                    id = 2;
                if (listBox2.SelectedItem == "Скай джоринг Пулка
спринт")
                    id = 3;
                if (listBox2.SelectedItem == "Пулка среднии
дистанции")
                    id = 4;
                if (listBox2.SelectedItem == "Драйленд кан кросс")
                    id = 5;
                if (listBox2.SelectedItem == "Лыжи - спринт")
                    id = 6;
                if (listBox2.SelectedItem == "Байк кросс")
                    id = 7;
                if (listBox2.SelectedItem == "Байк кросс")
```

```
                    id = 8;
                if (listBox2.SelectedItem == "Нарта - спринт")
                    id = 9;
                if (listBox2.SelectedItem == "Кросс")
                    id = 10;
                if (listBox2.SelectedItem == "Марафон")
                    id = 11;
                if (listBox2.SelectedItem == "Супер марафон")
                    id = 12;

                string command_string = "INSERT INTO `race` (ID_race,
Data_r, Time_r) VALUES (@id,@datar,@timer)";
                MySqlCommand mySqlCommand = new
MySqlCommand(command_string, db.GetConnection());
                mySqlCommand.Parameters.Add("@id",
MySqlDbType.Int32).Value = id;
                mySqlCommand.Parameters.Add("@datar",
MySqlDbType.Date).Value = dateTimePicker_r1.Value;
                mySqlCommand.Parameters.Add("@timer",
MySqlDbType.Date).Value = dateTimePicker_r2.Value;
                try
                {
                    db.OpenConnection();
                    if (mySqlCommand.ExecuteNonQuery() == 1)
                    {
                        MessageBox.Show("Заезд успешно добавлен",
"Успешно");
                    }
                    else
                    {
                        MessageBox.Show("Что-то пошло не так!",
"Ошибка!");
                    }
                }
                catch (Exception ex)
                {
                    MessageBox.Show(ex.ToString());
                }
                finally
                {
                    db.CloseConnection();
                }
            }

            public void loadData3()
            {
                database db = new database();
                string string_command = "select * from
`race`,`type_race`;";
                MySqlCommand mySqlCommand = new
MySqlCommand(string_command, db.GetConnection());
                try
                {
                    db.OpenConnection();
                    MySqlDataReader mySqlDataReader =
mySqlCommand.ExecuteReader();
```

```
                    dataGridView3.Columns.Add("ID_race", "Номер
гонки");
                    dataGridView3.Columns.Add("Data_r", "Начало
заезда");
                    dataGridView3.Columns.Add("Time_r", "Время
заезда");
                    dataGridView3.Columns.Add("Titles", "Тип гонки");
                    dataGridView3.Columns["ID_race"].Width = 100;
                    dataGridView3.Columns["ID_race"].ReadOnly = true;
                    dataGridView3.Columns["Data_r"].ReadOnly = true;
                    dataGridView3.Columns["Time_r"].ReadOnly = true;
                    dataGridView3.Columns["Titles"].ReadOnly = true;
                    while (mySqlDataReader.Read())
                    {

     dataGridView3.Rows.Add(mySqlDataReader["ID_race"].ToString(),
mySqlDataReader["Data_r"].ToString(),
mySqlDataReader["Time_r"].ToString(),
mySqlDataReader["Titles"].ToString());
                    }
               }
               catch (Exception)
               {
                    MessageBox.Show("Нет подключения к базе данных,
обратитесь к системному администратору.", "Ошибка!");
               }
               finally
               {
                    db.CloseConnection();
               }
          }

          private void button_del2_Click(object sender, EventArgs e)
          {
               database db = new database();
               string command_string = "DELETE FROM `type_race` WHERE
ID_race=@id";
               string command_string_1 = "DELETE FROM `race` WHERE
ID_race=@id";
               MySqlCommand sqlCommand = new
MySqlCommand(command_string, db.GetConnection());
               MySqlCommand sqlCommand_1 = new
MySqlCommand(command_string_1, db.GetConnection());
               sqlCommand.Parameters.Add("@id",
MySqlDbType.Int32).Value = textBox_rid.Text;
               try
               {
                    db.OpenConnection();
                    if (sqlCommand.ExecuteNonQuery() >= 1)
                    {
                         MessageBox.Show("Заезд успешно удален!",
"Успешно!");
                    }
                    else
                    {
                         MessageBox.Show("Заезда с таким артикулом не
существует.", "Ошибка!");
```

```
                    }
                }
                catch (Exception ex)
                {
                    MessageBox.Show(ex.ToString());
                    MessageBox.Show("Нет подключения к базе данных.
Обратитесь к администратору системы.", "Ошибка!");
                }
                finally
                {
                    db.CloseConnection();
                }
            }

            private void button_up3_Click(object sender, EventArgs e)
            {
                dataGridView3.Columns.Remove("ID_race");
                dataGridView3.Columns.Remove("Data_r");
                dataGridView3.Columns.Remove("Time_r");
                dataGridView3.Columns.Remove("Titles");
                loadData3();
            }

        private void button_addUser_Click(object sender, EventArgs e)
        {
                int role = 2;
                if (listBox1.SelectedItem == "Участник")
                    role = 0;
                else if (listBox1.SelectedItem == "Кинолог")
                    role = 1;
                else if (listBox1.SelectedItem == "Председатель")
                    role = 2;

                database db = new database();
                string command_string = "INSERT INTO `users` (login,
password, role) VALUES (@login,@password,@role)";
                MySqlCommand mySqlCommand = new
MySqlCommand(command_string, db.GetConnection());
                mySqlCommand.Parameters.Add("@login",
MySqlDbType.VarChar).Value = textBox_login.Text;
                mySqlCommand.Parameters.Add("@password",
MySqlDbType.VarChar).Value = textBox_passwd.Text;
                mySqlCommand.Parameters.Add("@role",
MySqlDbType.Int32).Value = role;
                try
                {
                    db.OpenConnection();
                    if (mySqlCommand.ExecuteNonQuery() == 1 )
                    {
                        MessageBox.Show("Пользователь успешно
добавлен", "Успешно");
                    }
                    else
                    {
                        MessageBox.Show("Что-то пошло не так!",
"Ошибка!");
                    }
```

```
                }
                catch (Exception ex)
                {
                      MessageBox.Show(ex.ToString());
                }
                finally
                {
                      db.CloseConnection();
                }

            }

        private void button_delUser_Click(object sender, EventArgs e)
        {
                database db = new database();
                string command_string = "DELETE FROM `users` WHERE
user_id=@id";
                MySqlCommand sqlCommand = new
MySqlCommand(command_string, db.GetConnection());
                sqlCommand.Parameters.Add("@id",
MySqlDbType.Int32).Value = textBox_uid.Text;
                try
                {
                      db.OpenConnection();
                      if (sqlCommand.ExecuteNonQuery() >= 1)
                      {
                            MessageBox.Show("Пользователь успешно
удален!", "Успешно!");
                      }
                      else
                      {
                            MessageBox.Show("Пользователя с таким
артикулом не существует.", "Ошибка!");
                      }
                }
                catch (Exception ex)
                {
                      MessageBox.Show(ex.ToString());
                      MessageBox.Show("Нет подключения к базе данных.
Обратитесь к администратору системы.", "Ошибка!");
                }
                finally
                {
                      db.CloseConnection();
                }
            }

            public void loadData4()
            {
                  database db = new database();
                  string string_command = "select * from `users`;";
                  MySqlCommand mySqlCommand = new
MySqlCommand(string_command, db.GetConnection());
                  try
                  {
                        db.OpenConnection();
```

```
                    MySqlDataReader mySqlDataReader =
mySqlCommand.ExecuteReader();
                    dataGridView4.Columns.Add("user_id",
"Идентификатор пользователя");
                    dataGridView4.Columns.Add("login", "Имя
пользователя");
                    dataGridView4.Columns.Add("password", "Пароль");
                    dataGridView4.Columns.Add("role", "Роль");
                    dataGridView4.Columns["user_id"].Width = 100;
                    dataGridView4.Columns["user_id"].ReadOnly = true;
                    dataGridView4.Columns["login"].ReadOnly = true;
                    dataGridView4.Columns["password"].ReadOnly =
true;
                    dataGridView4.Columns["role"].ReadOnly = true;
                    while (mySqlDataReader.Read())
                    {

     dataGridView4.Rows.Add(mySqlDataReader["user_id"].ToString(),
mySqlDataReader["login"].ToString(),
mySqlDataReader["password"].ToString(),
mySqlDataReader["role"].ToString());
                    }
                }
                catch (Exception)
                {
                    MessageBox.Show("Нет подключения к базе данных,
обратитесь к системному администратору.", "Ошибка!");
                }
                finally
                {
                    db.CloseConnection();
                }
            }

            private void button_up4_Click(object sender, EventArgs e)
        {
                dataGridView4.Columns.Remove("user_id");
                dataGridView4.Columns.Remove("login");
                dataGridView4.Columns.Remove("password");
                dataGridView4.Columns.Remove("role");
                loadData4();
            }
    }
}
Листинг подсистемы кинолог
using System;
using System.Collections.Generic;
```

Листинг Б.2 – подсистемы Председатель на языке C# (файл kin_panel .cs)

```
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
```

```
using MySql.Data.MySqlClient;
namespace dogs
{
    public partial class kin_panel : Form
    {
        public kin_panel()
        {
            InitializeComponent();
        }

        private void button5_Click(object sender, EventArgs e)
        {
                int id = 0;
                if (listBox1.SelectedItem == "Хаски")
                     id = 1;
                if (listBox1.SelectedItem == "Маламут")
                     id = 2;
                if (listBox1.SelectedItem == "Самоедская лайка")
                     id = 3;
                if (listBox1.SelectedItem == "Сибо ино")
                     id = 4;
                if (listBox1.SelectedItem == "Гренландская ездовая
собака")
                     id = 5;
                if (listBox1.SelectedItem == "Норвежский спортивный
метис")
                     id = 6;
                if (listBox1.SelectedItem == "Еврохаунд")
                     id = 7;
                if (listBox1.SelectedItem == "Другая")
                     id = 8;

                database db = new database();

                string command_string = "INSERT INTO `dogs` (ID_Book,
ID_Breed, ID_pass , ID_mark_dog, Breeder, Name_dog, Year_dog) VALUES
(@ID_Book,
@ID_Breed,@ID_pass,@ID_mark_dog,@Breeder,@Name_dog,@Year_dog)";
                MySqlCommand mySqlCommand = new
MySqlCommand(command_string, db.GetConnection());
                mySqlCommand.Parameters.Add("@ID_Book",
MySqlDbType.Int32).Value = textBox_id.Text;
                mySqlCommand.Parameters.Add("@ID_Breed",
MySqlDbType.Int32).Value = textBox_num.Text;
                mySqlCommand.Parameters.Add("@ID_pass",
MySqlDbType.Int32).Value = textBox_passport;
                mySqlCommand.Parameters.Add("@ID_mark_dog",
MySqlDbType.Int32).Value = id;
                mySqlCommand.Parameters.Add("@Breeder",
MySqlDbType.VarChar).Value = breed.Text;
                mySqlCommand.Parameters.Add("@Name_dog",
MySqlDbType.VarChar).Value = textBox_name.Text;
                mySqlCommand.Parameters.Add("@Year_dog",
MySqlDbType.Date).Value = dateTimePicker1.Value;

                try
                {
```

```
                    db.OpenConnection();
                    if (mySqlCommand.ExecuteNonQuery() == 1)
                    {
                        MessageBox.Show("Собака успешно добавлена",
"Успешно");
                    }
                    else
                    {
                        MessageBox.Show("Что-то пошло не так!",
"Ошибка!");
                    }
                }
                catch (Exception ex)
                {
                    MessageBox.Show(ex.ToString());
                }
                finally
                {
                    db.CloseConnection();
                }
            }

        private void button7_Click(object sender, EventArgs e)
        {
                int id = 0;
                if (listBox2.SelectedItem == "Прививка")
                    id = 1;
                if (listBox2.SelectedItem == "Осмотр молодняка")
                    id = 2;
                if (listBox2.SelectedItem == "Травма общая")
                    id = 3;
                if (listBox2.SelectedItem == "Травма конечности")
                    id = 4;
                if (listBox2.SelectedItem == "Инфекционное
заболевание")
                    id = 5;
                if (listBox2.SelectedItem == "Щенность")
                    id = 6;
                if (listBox2.SelectedItem == "Вязка")
                    id = 7;
                if (listBox2.SelectedItem == "Биометрия")
                    id = 8;
                if (listBox2.SelectedItem == "Операционное
вмещательство")
                    id = 9;

                database db = new database();

                string command_string = "INSERT INTO
`med_passport_dog` (ID_pass, ID_string) VALUES (@ID_pass,
@ID_string)";
                MySqlCommand mySqlCommand = new
MySqlCommand(command_string, db.GetConnection());
                mySqlCommand.Parameters.Add("@ID_pass",
MySqlDbType.Int32).Value = textBox9.Text;
                mySqlCommand.Parameters.Add("@ID_string",
MySqlDbType.Int32).Value = id;
```

```
                try
                {
                      db.OpenConnection();
                      if (mySqlCommand.ExecuteNonQuery() == 1)
                      {
                            MessageBox.Show("Мед.запись успешно
добавлена", "Успешно");
                      }
                      else
                      {
                            MessageBox.Show("Что-то пошло не так!",
"Ошибка!");
                      }
                }
                catch (Exception ex)
                {
                      MessageBox.Show(ex.ToString());
                }
                finally
                {
                      db.CloseConnection();
                }
           }

        private void button8_Click(object sender, EventArgs e)
        {
                int id = 0;
                if (listBox3.SelectedItem == "Чемпион России")
                      id = 1;
                if (listBox3.SelectedItem == "Юный чемпион России")
                      id = 2;
                if (listBox3.SelectedItem == "Чемпион федерации")
                      id = 3;
                if (listBox3.SelectedItem == "Чемпион российской
кинологической федерации")
                      id = 4;
                if (listBox3.SelectedItem == "Гран чемпион России")
                      id = 5;
                if (listBox3.SelectedItem == "Чемпион национального
кубка")
                      id = 6;
                if (listBox3.SelectedItem == "Юный чемпион
национального кубка")
                      id = 7;
                if (listBox3.SelectedItem == "Интернациональный
чемпион")
                      id = 8;
                if (listBox3.SelectedItem == "Мультичемпион")
                      id = 9;
                if (listBox3.SelectedItem == "Чемпион пароды года")
                      id = 10;
                if (listBox3.SelectedItem == "Собака года")
                      id = 11;
                if (listBox3.SelectedItem == "Элита")
                      id = 12;
```

```
                database db = new database();

                string command_string = "INSERT INTO `dog_rewards`
(ID_mark_dog, ID_totaly,ID_race,Награда) VALUES (@ID_mark_dog,
@ID_totaly,@ID_race,@award)";
                MySqlCommand mySqlCommand = new
MySqlCommand(command_string, db.GetConnection());
                mySqlCommand.Parameters.Add("@ID_mark_dog",
MySqlDbType.VarChar).Value = textBox11.Text;
                mySqlCommand.Parameters.Add("@ID_totaly",
MySqlDbType.Int32).Value = id;
                mySqlCommand.Parameters.Add("@ID_race",
MySqlDbType.VarChar).Value = textBox8.Text;
                mySqlCommand.Parameters.Add("@award",
MySqlDbType.VarChar).Value = textBox1.Text;
                try
                {
                      db.OpenConnection();
                      if (mySqlCommand.ExecuteNonQuery() == 1)
                      {
                            MessageBox.Show("Мед.запись успешно
добавлена", "Успешно");
                      }
                      else
                      {
                            MessageBox.Show("Что-то пошло не так!",
"Ошибка!");
                      }
                }
                catch (Exception ex)
                {
                      MessageBox.Show(ex.ToString());
                }
                finally
                {
                      db.CloseConnection();
                }
           }
        private void button9_Click(object sender, EventArgs e)
        {

        }  }}
```

Приложение В
ЛИСТИНГ КЛАССОВ БАЗЫ ДАННЫХ
(рекомендуемое)

```
CREATE TABLE 17.Breed
(
      ID_breed int  NOT NULL ,
      Title_breed varchar(18)  NULL
)
go

ALTER TABLE 17.Breed
      ADD  PRIMARY KEY (ID_breed ASC)
go
CREATE TABLE 17.Dog_rewards
(
      ID__кmark_dog varchar(18)  NOT NULL ,
      ID_totaly int  NOT NULL ,
      ID__race int  NOT NULL ,
      Награда varchar(18)  NULL
)
go
ALTER TABLE 17.Dog_rewards
      ADD  PRIMARY KEY (ID__кmark_dog ASC,ID_totaly ASC,ID__race ASC)
go
CREATE TABLE 17.Dogs
(
      ID_book int  NOT NULL ,
      ID_breed int  NOT NULL ,
      ID_pass int  NOT NULL ,
      ID_mark_dog int  NOT NULL ,
      Breeder varchar(18)  NULL ,
      Name_dog varchar(18)  NOT NULL ,
      Year_dog datetime  NULL
)
go
ALTER TABLE 17.Dogs
      ADD  PRIMARY KEY (ID_book ASC,ID_breed ASC,ID_pass
ASC,ID_mark_dog ASC)
go
CREATE TABLE 17.In_races
(
      ID_mark_dog int  NOT NULL ,
      ID__sled int  NOT NULL ,
      ID_team int  NOT NULL ,
      ID__race int  NOT NULL ,
      Number_r integer  NOT NULL
)
go
LTER TABLE 17.In_races
      ADD  PRIMARY KEY (ID_mark_dog ASC,ID_team ASC,ID__race
ASC,ID__sled ASC)
go
CREATE TABLE 17.Med_passport__dog
(
      ID_pass int  NOT NULL ,
      ID_string int  NOT NULL ,
      Text varchar(50)  NULL
)
go
ALTER TABLE 17.Med_passport__dog
```

```
      ADD  PRIMARY KEY (ID_pass ASC)
go
CREATE TABLE 17.Owners
(
      ID_book int  NOT NULL ,
      FIO varchar(30)  NOT NULL ,
      Address varchar(40)  NULL ,
      Year_birth datetime  NOT NULL ,
      Year_entry datetime  NULL
)
go
ALTER TABLE 17.Owners
      ADD  PRIMARY KEY (ID_book ASC)
go

CREATE TABLE 17.Payment
(
      ID_book int  NOT NULL ,
      Account_number int  NOT NULL ,
      Data_p datetime  NOT NULL ,
      Moon varchar(10)  NOT NULL ,
      Summa int  NOT NULL ,
      App_Not binary(2)  NOT NULL
)
go
ALTER TABLE 17.Payment
      ADD  PRIMARY KEY (ID_book ASC)
go
CREATE TABLE 17.Race
(
      ID__race int  NOT NULL ,
      Data_r datetime  NULL ,
      Tyme_r datetime  NULL
)
go
ALTER TABLE 17.Race
      ADD  PRIMARY KEY (ID__race ASC)
go

CREATE TABLE 17.Totaly
(
      ID_totaly int  NOT NULL ,
      Титул varchar(18)  NULL
)
go
ALTER TABLE 17.Totaly
      ADD  PRIMARY KEY (ID_totaly ASC)
go
CREATE TABLE 17.Type_of_team
(
      ID_team int  NOT NULL ,
      Quantity int  NULL
)
go
ALTER TABLE 17.Type_of_team
      ADD  PRIMARY KEY (ID_team ASC)
go
```

```
CREATE TABLE 17.Type_race
(
      ID__race int  NOT NULL ,
      Titles varchar(18)  NULL
)
goALTER TABLE 17.Type_race
      ADD  PRIMARY KEY (ID__race ASC)
go


CREATE TABLE 17.Type_sled
(
      ID__sled int  NOT NULL ,
      Title varchar(18)  NULL ,
      Netto int  NULL
)
go

ALTER TABLE 17.Type_sled
      ADD  PRIMARY KEY (ID__sled ASC)
go

CREATE TABLE 17.Type_string
(
      ID_string int  NOT NULL ,
      String varchar(18)  NULL
)
go
ALTER TABLE 17.Type_string
      ADD  PRIMARY KEY (ID_string ASC)
Go
create table Owners(
      ID_book int NOT NULL primary key,
      FIO varchar(30) NOT NULL,
      Address varchar(4) NULL,
      Year_birth datetime NOT NULL,
      Year_entry datetime NULL
      );

create table Payment(
      ID_book int,
      foreign key (ID_book) references Owners (ID_book),
    Account_number int NOT NULL,
    Data_p datetime NOT NULL,
    Moon varchar(10) NOT NULL,
    Summa int NOT NULL,
    App_Not binary(2) NOT NULL
    );

create table Breed(
      ID_breed int NOT NULL primary key,
    Title_breed int NOT NULL
    );
```

```
create table Type_string(
     ID_string int primary key NOT NULL,
    String varchar(18) null
    );

create table Med_passport_dog(
     ID_pass int NOT NULL primary key,
    ID_string int,
     foreign key (ID_string) references Type_string (ID_string)
    );

create table Dogs(
     ID_book int,
    ID_breed int,
    ID_pass int,
     foreign key (ID_book) references Owners (ID_book),
    foreign key (ID_breed) references Breed (ID_breed),
    foreign key (ID_pass) references Med_passport_dog (ID_pass),
    ID_mark_dog int primary key NOT NULL,
    Breeder varchar(18) NULL,
    Name_dog varchar(18) NOT NULL,
    Year_dog datetime NULL
    );

create table Totaly(
     ID_totaly int primary key NOT NULL,
    Титул varchar(18) NULL
    );

create table Type_race(
     ID_race int primary key NOT NULL,
    Titles varchar(18) NULL
    );

create table Race(
     ID_race int,
     foreign key (ID_race) references Type_race (ID_race),
    Data_r datetime NULL,
    Time_r datetime NULL
    );

create table Dog_rewards(
     ID_mark_dog int,
    ID_totaly int,
    ID_race int,
     foreign key (ID_mark_dog) references Dogs (ID_mark_dog),
    foreign key (ID_totaly) references Totaly (ID_totaly),
    foreign key (ID_race) references Type_race (ID_race),
```

```
    Награда varchar(18) NULL
    );

create table Type_sied(
     ID_sied int primary key NOT NULL,
    Title varchar(18) NULL,
    Netto int NULL
    );

create table Type_of_team(
     ID_team int primary key NOT NULL,
    Quantity int NULL
    );

create table In_races(
     ID_mark_dog int,
    ID_team int,
    ID_race int,
    ID_sied int,
     foreign key (ID_mark_dog) references Dogs (ID_mark_dog),
    foreign key (ID_team) references Type_of_team (ID_team),
    foreign key (ID_race) references Type_race (ID_race),
    foreign key (ID_sied) references Type_sied (ID_sied),
    Number_r integer NOT NULL
    );
```

Приложение Г
РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ
(рекомендуемое)

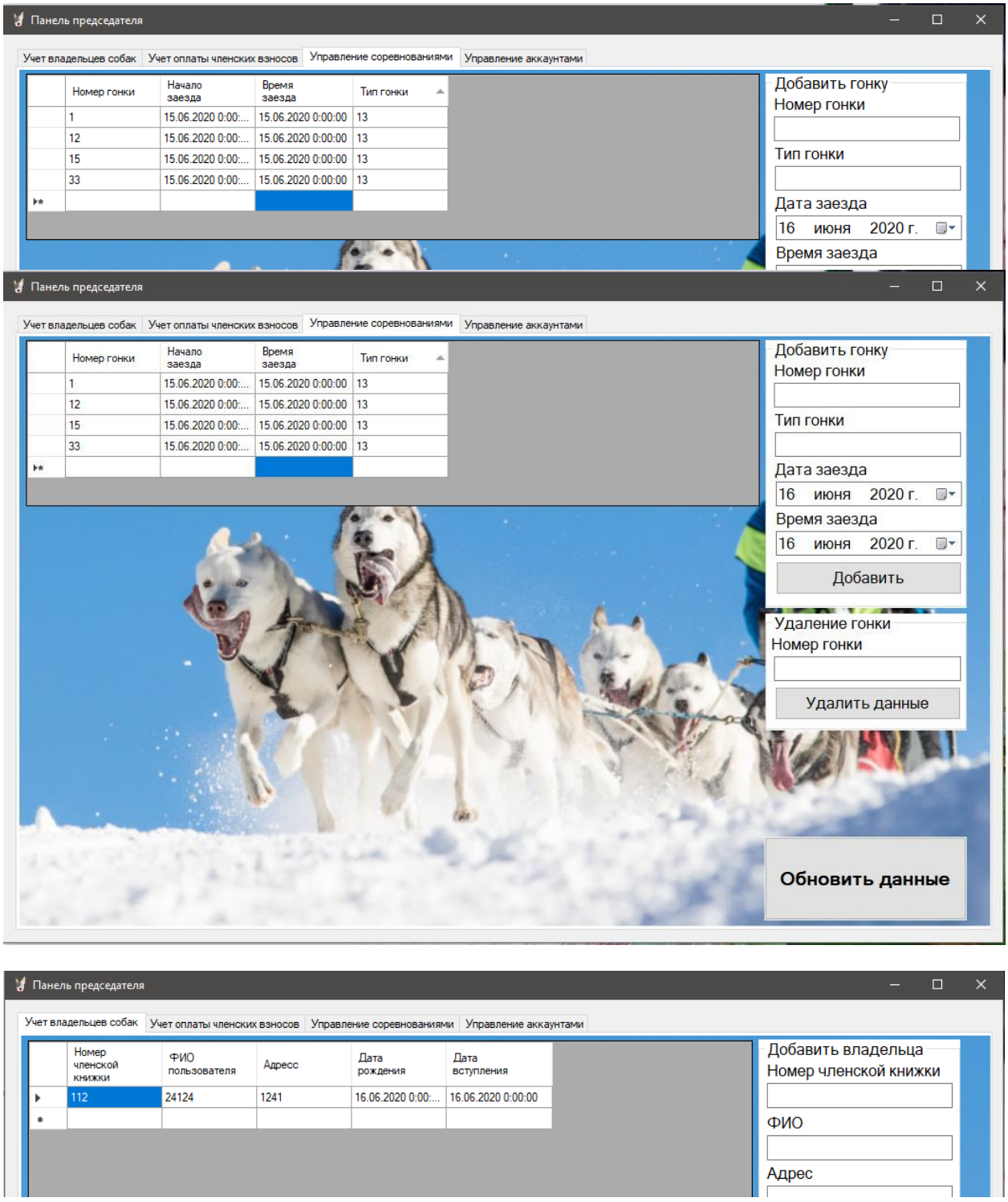


Рисунок Г.1 – Вход в систему, подсистема председатель (тест 1)

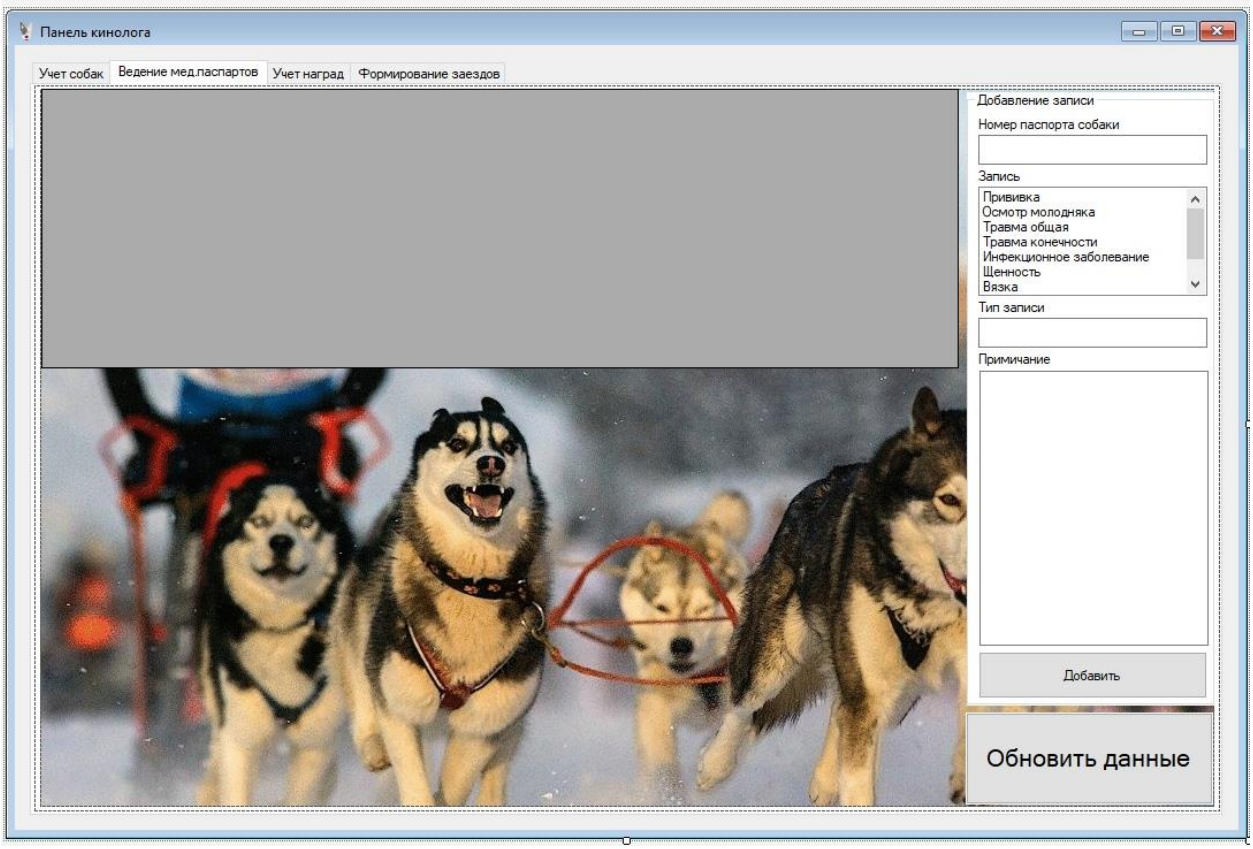


Рисунок Г.2 – Вход в систему, подсистема кинолог (тест 2)

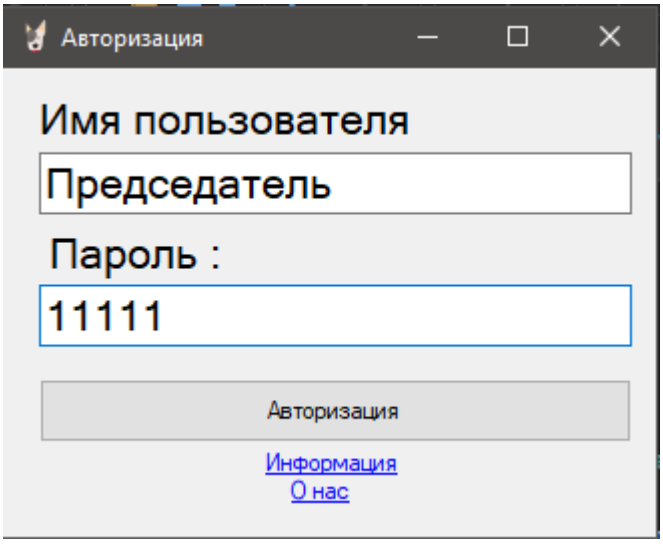


Рисунок Г.3 – Вход в систему (тест 3)

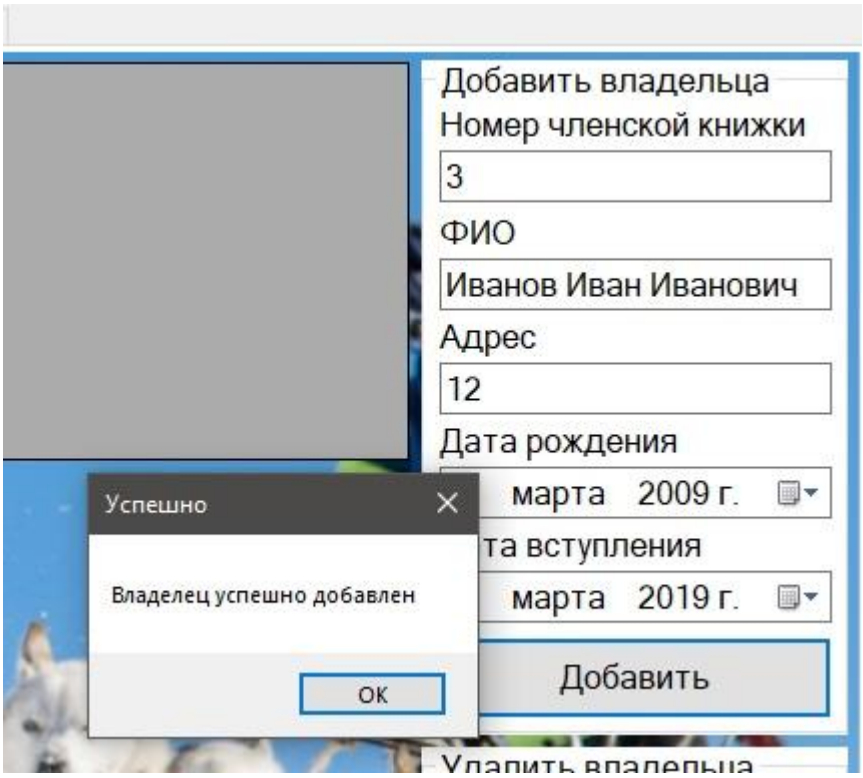


Рисунок Г.4 – Добавить владельца собаки (тест 4)

Приложение Д
ЛИСТИНГ ТЕСТОВОГО ПРОЕКТА
(рекомендуемое)

Листинг Д.1 – Тестовый проект на языке C# (файл UnitTest1.cs)

```
using System;
using libraryFond;
using Microsoft.VisualStudio.TestTools.UnitTesting;
using LibraryApp;

namespace LIbTests
{
	[TestClass]
	public class ReaderRegTest
	{
		[TestMethod]
		public void RegTrueTest()
		{
			var readerReg = new reader_reg();
			int res = readerReg.reg("Панфилов", "Пётр",
				"Анатольевич", Convert.ToDateTime("01/01/1998"),
				"5555-986547","г.Москва,ул.Манежная,20",
				"+7(564)-584-35-15", 532);
			Assert.AreEqual(0, res);
		}

		[TestMethod]
		public void FalseFamTest()
		{
			var readerReg = new reader_reg();
			int res = readerReg.reg("", "Пётр", "Анатольевич",
				Convert.ToDateTime("01/01/1998"), "5555-986547",
				"г.Москва,ул.Манежная,20", "+7(564)-584-35-15", 532);
			Assert.AreEqual(1, res);
		}

		[TestMethod]
		public void FalseNameTest()
		{
			var readerReg = new reader_reg();
			int res = readerReg.reg("Панфилов", "",
				"Анатольевич", Convert.ToDateTime("01/01/1998"),
				"5555-986547",
				"г.Москва,ул.Манежная,20", "+7(564)-584-35-15", 532);
			Assert.AreEqual(1, res);
		}

		[TestMethod]
		public void FalsePatroTest()
		{
			var readerReg = new reader_reg();
			int res = readerReg.reg("Панфилов", "Пётр", "",
				Convert.ToDateTime("01/01/1998"), "5555-986547",
				"г.Москва,ул.Манежная,20", "+7(564)-584-35-15", 532);
			Assert.AreEqual(1, res);
		}

		[TestMethod]
		public void FalsePassportTest()
		{
			var readerReg = new reader_reg();
			int res = readerReg.reg("Панфилов", "Пётр",
				"Анатольевич", Convert.ToDateTime("01/01/1998"),
				"","г.Москва,ул.Манежная,20","+7(564)-584-35-15",
				532);
			Assert.AreEqual(1, res);
```

```
        }
    }

    [TestClass]
    public class ReaderUpdateTest
    {
        [TestMethod]
        public void UpdateTrueTest()
        {
            var readerReg = new reader_update();
            int res = readerReg.update(1034, "Панфилов", "Пётр",
                "Анатольевич", Convert.ToDateTime("01/01/1998"),
                "5555-986547",  "г.Москва,ул.Манежная,20",  "+7(564)-
                584-35-15", 532);
            Assert.AreEqual(0, res);
        }

        [TestMethod]
        public void UpdateFalseFamTest()
        {
            var readerReg = new reader_update();
            int res = readerReg.update(1034, "", "Пётр",
                "Анатольевич", Convert.ToDateTime("01/01/1998"),
                "5555-986547","г.Москва,ул.Манежная,20",
                "+7(564)-584-35-15", 532);
            Assert.AreEqual(1, res);
        }
        [TestMethod]
        public void UpdateFalseNameTest()
        {
            var readerReg = new reader_update();
            int res = readerReg.update(1034, "Панфилов", "",
                "Анатольевич", Convert.ToDateTime("01/01/1998"),
                "5555-986547", "г.Москва,ул.Манежная,20",
                "+7(564)-584-35-15", 532);
            Assert.AreEqual(1, res);
        }

        [TestMethod]
        public void UpdateFalsePatroTest()
        {
            var readerReg = new reader_update();
            int res = readerReg.update(1034, "Панфилов", "Пётр",
                "",Convert.ToDateTime("01/01/1998"),"5555-986547",
                "г.Москва,ул.Манежная,20", "+7(564)-584-35-15", 532);
            Assert.AreEqual(1, res);
        }

        [TestMethod]
        public void UpdateFalsePassportTest()
        {
            var readerReg = new reader_update();
            int res = readerReg.update(1034, "Панфилов", "Пётр",
                "Анатольевич", Convert.ToDateTime("01/01/1998"),
                "","г.Москва,ул.Манежная,20", "+7(564)-584-35-15",
                532);
            Assert.AreEqual(1, res);
        }
    }
    [TestClass]
    public class BookRegTest
    {
```

```
        [TestMethod]
              public void RegTrueTest()
              {
                    var OwnerDogsReg = new OnwerDogs_Reg();
                    int res =OnwerDogs.reg("3", "Иванов Иван Иванович",,
                    "Столбовая 12", Convert.ToDateTime("01/04/2009"),
                    Convert.ToDateTime("01/04/2019"),
                    Assert.AreEqual(0, res);
              }
        [TestMethod]
              public void FalseIDTest()
              {
                    var OwnerDogsReg = new OnwerDogs_Reg();
                    int res = DogsReg.reg("", " Иванов Иван Иванович ",
                    , "Столбовая 12", Convert.ToDateTime("01/04/2009"),
                    Convert.ToDateTime("01/04/20109"),
                    Assert.AreEqual(1, res);
              }

        [TestMethod]
              public void FalseFIOTest()
              {
                    var OwnerDogsReg = new OnwerDogs_Reg();
                     int res = DogsReg.reg("3", "",
                      , "Столбовая 12", Convert.ToDateTime("01/04/2009"),
                     Convert.ToDateTime("01/04/2019"),
                     Assert.AreEqual(1, res);
              }
        [TestMethod]
              public void FalseAddressTest()
              {
                    var OwnerDogsReg = new OnwerDogs_Reg();
                     int res = DogsReg.reg("3", " Иванов Иван Иванович ",
                     , "", Convert.ToDateTime("01/04/2009"),
                     Convert.ToDateTime("01/04/2019"),
                     Assert.AreEqual(1, res);
              }
        [TestMethod]
              public void FalseDateOfBirthTest()
              {
                    var OwnerDogsReg = new OnwerDogs_Reg();
                    int res = DogsReg.reg("3", " Иванов Иван Иванович ",
                    , "Столбовая 12", Convert.ToDateTime(""),
                    Convert.ToDateTime("01/04/2019"),
                    Assert.AreEqual(1, res);
              }
        [TestMethod]
              public void FalseDateOfEntryTest()
              {
                    var OwnerDogsReg = new OnwerDogs_Reg();
                    int res = DogsReg.reg("3", " Иванов Иван Иванович ",
                    , "Столбовая 12", Convert.ToDateTime("01/04/2009"),
                    Convert.ToDateTime(""),
                    Assert.AreEqual(1, res);
              }
        }
}
```