

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение высшего  
образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ  
И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра безопасности информационных систем (БИС)

**АИС ДЛЯ ПРЕДМЕТНОЙ ОБЛАСТИ «КАЗИНО»**

Курсовая работа по дисциплине  
«Безопасность систем баз данных»

Пояснительная записка

Студент группы 721-1

\_\_\_\_\_ Козлов К.М.

«\_\_» \_\_\_\_\_ 2024г.

Руководитель

Ст. преподаватель каф. КИБЭВС

\_\_\_\_\_ Н.А. Новгородова

«\_\_» \_\_\_\_\_ 2024г.

## Реферат

Курсовая работа содержит 74 страницы, 38 рисунков, 11 таблиц, 7 источников.

БАЗА ДАННЫХ SQL, SSMS СУБД, АИС WINDOWS FORMS, C#.

АИС для автоматизации процесса игры в покер, связанная с базой данных. Целью данного проекта является разработка и настройка базы данных для автоматизации игры в покер в казино. В рамках курсовой работы была спроектирована архитектура базы данных в Microsoft SQL Server Management Studio (SSMS) для хранения информации о игроках, их ставках, крупье, а также о текущем состоянии игры. После чего было применено ролевое разграничение, а также была создана собственная АИС.

Разработка АИС была произведена на языке программирования C# в Windows Form.

Курсовая работа выполнена в текстовом редакторе Microsoft Word 2016. Пояснительная записка оформлена согласно ОС ТУСУР 01-2021 [1].

## **The abstract**

The term paper contains 74 pages, 38 figures, 11 tables, 7 sources.

DATABASE SQL, SSMS DBMS, AIS WINDOWS FORM C#.

AIS for automating the poker game process associated with the database.

The purpose of this project is to develop and configure a database for automating poker games in casinos. As part of the course work, a database architecture was designed in Microsoft SQL Server Management Studio (SSMS) to store information about players, their bets, croupiers, as well as the current state of the game. After that, role differentiation was applied, and its own AIS was created.

The AIS was developed in the C# programming language in Windows Form. The course work was done in a Microsoft Word 2016 text editor.

The explanatory note is issued according to OS TUSUR 01-2021 [1].

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение высшего  
образования  
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ  
И РАДИОЭЛЕКТРОНИКИ (ТУСУР)  
Кафедра безопасности информационных систем (БИС)

АИС ДЛЯ ПРЕДМЕТНОЙ ОБЛАСТИ «Казино»  
ЗАДАНИЕ

СОГЛАСОВАНО

Ст. преподаватель каф.

КИБЭВС

\_\_\_\_ Н.А.Новгородова

\_\_\_\_\_ 2023г.

РАЗРАБОТЧИК

Студент гр. 741-1

\_\_\_\_\_ Козлов К.М.

\_\_\_\_\_ 2023г.

На курсовую работу по дисциплине «Безопасность систем баз данных» студенту факультета безопасности группы 721-1 Козлову Кириллу Максимовичу.

1 Тема работы: Автоматизированная информационная система с базой данных: АИС ДЛЯ ПРЕДМЕТНОЙ ОБЛАСТИ «КАЗИНО».

2 Исходные данные к работе:

2.1 Реляционная СУБД: Microsoft SQL Server;

2.2 Данные по предметной области: предметная область «Казино»

3 Срок сдачи студентом законченной работы: 30.01.2024г.

4 Содержание курсовой работы:

4.1 Проектирование инфологической модели данных:

- описание и структуризация предметной области (описание бизнес-процессов, диаграммы IDEF0);
- представление модели «Сущность-связь» (ER-модель);
- сценарийпользовательского интерфейса.

4.2 Проектирование даталогической (логической) модели данных:

- проектирование реляционной базы данных на основе принциповнормализации;
- проектирование концептуальной модели данных (использованиеметодологии IDEF1X);
- составление глоссария модели.

4.3 Физическое проектирование БД:

- создание базы данных и ее необходимых элементов;
- описание ограничений на базу данных;
- сопоставление логических и физических имен.

#### 4.4 Вопрос безопасности данных БД;

#### 4.5 Написание программы обработки и работы с данными:

- генерация программы меню, реализующей пользовательский интерфейс;
- режим просмотра данных с использованием экранных форм;
- использование режимов редактирования данных;
- процедуры поиска и манипулирования данными (сортировки, фильтры и пр.);
- использование SQL операторов (SQL запросы, операторы определения данных, операторы манипулирования данными);
- обеспечение безопасности данных.

#### 4.6 Вопросы конфиденциальности данных БД;

### 5 Содержание пояснительной записки:

- титульный лист;
- реферат на русском языке;
- реферат на английском языке;
- задание;
- оглавление;
- введение;
- обоснование выбора программных средств;
- вопросы проектирования БД;
- описание прикладной программы;
- обеспечение безопасности ПДН;
- заключение;
- список использованных источников;
- приложения (экранные формы, листинг программы и др.).

Пояснительная записка должна быть оформлена в соответствии со стандартом ТУСУР.

6 Дата выдачи задания: «\_\_\_»\_\_\_\_\_2023 г.

СОГЛАСОВАНО

ПРИНЯЛ К ИСПОЛНЕНИЮ

Ст. преподаватель каф. КИБЭВС

Студент гр. 721-1

\_\_\_\_\_ Новгородова Н.А

\_\_\_\_\_ Козлов К.М.

\_\_\_\_\_ 2023г.

\_\_\_\_\_ 2023г.

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение высшего  
образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ  
И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра безопасности информационных систем (БИС)

АИС ДЛЯ ПРЕДМЕТНОЙ ОБЛАСТИ «КАЗИНО»

Техническое задание

На 10 листах

СОГЛАСОВАНО

Ст. преподаватель каф.

КИБЭВС

\_\_\_\_\_ Новгородова Н.А.

\_\_\_\_\_ 2023г.

РАЗРАБОТЧИК

Студент гр. 721-1

\_\_\_\_\_ Козлов К.М.

\_\_\_\_\_ 2023г.



## 1 Общие сведения

### 1.1 Полное наименование системы и ее условное обозначение

Полное наименование системы: «АИС ДЛЯ ПРЕДМЕТНОЙ ОБЛАСТИ «КАЗИНО».

### 1.2 Заказчик

Заказчиком является Томский государственный университет систем управления и радиоэлектроники (ТУСУР), кафедра безопасности информационных систем (БИС).

### 1.3 Исполнитель

Исполнителем является студент группы 721-1 Козлов Кирилл Максимович.

### 1.4 Основания разработки

Основанием для разработки является задание на выполнение курсовой работы по дисциплине «Безопасность систем баз данных» для студентов направления подготовки 10.05.03 – «Информационная безопасность автоматизированных систем».

## 2 Назначение и цель создания системы

### 2.1 Назначение системы

Система предназначена для работы со специализированной базой данных от имени различных ролей для предметной области, связанной с работой казино.

### 2.2 Цель создания системы

Целью разработки является автоматизация процесса работы от имени различных ролей для предметной области, связанной с работой казино.

### 3 Характеристика объектов автоматизации

#### 3.1 Объект автоматизации

Объектом автоматизации является процесс работы со специализированной базой данных от имени различных ролей для предметной области, связанной с работой казино.

## 4 Требования к системе

### 4.1 Требования к структуре и функционированию

Приложение должно выполнять следующие функции:

- Авторизация для пользователей всех типов ролей;
- Регистрация пользователей при помощи SSMS;
- Просмотр и работа с таблицами на уровне обозначенных прав;

### 4.2 Перечень подсистем, их назначение и основные характеристики

В системе предлагается выделить следующие функциональные подсистемы:

- Подсистема авторизации;
- Подсистема работы с базами данных.

### 4.3 Требования к надёжности

При возникновении сбоев в аппаратном обеспечении, включая разряд аккумулятора устройства, информационная система восстанавливает свою работоспособность после устранения сбоев и корректного перезапуска программного обеспечения (за исключением случаев повреждения рабочих носителей информации с исполняемым программным кодом).

### 4.4 Требования к безопасности

Все технические решения, использованные при создании системы, а также при определении требований к аппаратному обеспечению, соответствуют действующим нормам и правилам техники безопасности, пожарной безопасности, а также охраны окружающей среды при эксплуатации.

#### 4.5 Требования к эксплуатации, техническому обслуживанию, ремонту, хранению

Для эксплуатации разрабатываемой информационной системы необходимы следующие условия:

- Компьютер под управлением операционной системы Windows 11/10/8/8.1/7;
- Наличие таких периферийных устройств, как мышь и клавиатура, для взаимодействия;
- Питание компьютера от сети или батареи.

#### 4.6 Требования к защите информации от несанкционированного доступа

Доступ к работе с интерфейсом системы имеют все пользователи, доступ к работе с внутренними модулями программы имеют только разработчики.

#### 4.7 Требования к функциям разработчика

Роль разработчика заключается в обновлении и пополнении системы новыми функциями, а также исправление возможных ошибок в функционировании системы. Так же разработчику будет предоставлены полные права для работы с базой данных.

#### 4.8 Требования к функциям пользователя

При использовании приложения пользователь может авторизоваться или запросить регистрацию учетной записи у системного администратора. Также пользователь может на уровне своих прав произвести работу с базой данных. Пользователю в зависимости от прав доступны такие функции как: обновление данных, удаление данных, добавление данных, просмотр данных.

#### 4.9 Описание процессов и функций работы с системой

Выполняемые при эксплуатации системы процессы и функции приведены в разбивке по подсистемам: подсистема авторизации, подсистема регистрации, подсистема работы с базой данных.

Процессы, реализованные под управлением различных подсистем, реализуются на основе системных процедур, которые являются составной частью функций системы. Системные процедуры группируются в соответствии с их назначением:

- авторизация;
- работа с базой данных.

#### 4.10 Требования к информационному обеспечению системы

Информационное обеспечение должно быть достаточным для поддержания всех автоматизируемых функций объекта. Должна быть обеспечена совместимость с информационным обеспечением систем, взаимодействующих с разрабатываемой системой. В ИС должны быть предусмотрены средства контроля входной и результатной информации, обновления данных в информационных массивах, контроля целостности информационной базы, защиты от несанкционированного доступа.

#### 4.11 Требования к программному обеспечению

- ОС Windows 11/10/8.1/8/7;
- СУБД Microsoft SQL Server;
- SQL Server Management Studio от Microsoft;
- Языки программирования C# и SQL.

## 5 Состав и содержание работ по созданию системы

Состав и содержание работ по созданию системы приведены в таблице 5.1.

Таблица 5.1 - Этапы разработки

№	Этап	Результат	Срок выполнения
1	Выбор предметной области для курсовой работы	Выбранная предметная область	10.09.2023г.
2	Моделирование базы данных	Модель базы данных	25.09.2023г.
4	Выбор средств для проектирования базы данных	Средства разработки, удовлетворяющие условиям проектирования базы данных	15.10.2023г.
5	Проектирование базы данных и заполнение данных в ней	Спроектированная база данных и набор данных для нее	30.10.2023г.
6	Выбор средств разработки десктопного приложения	Средства разработки, удовлетворяющие условиям реализации десктопного приложения	15.11.2023г.
7	Разработка десктопного приложения и его связьс базой данных	Разработанное приложение готовое для работы с базой данных	20.11.2023г.
8	Защита курсовой работы	Защищенная курсовая работа	01.02.2024г.

## 6 Перечень контроля и приемки системы

### 6.1 Перечень этапов испытаний и проверок

Этапы испытаний подразделяются на предварительные и приемочные. Предварительные испытания проводятся на стадии тестирования разработчиком. Приемочные испытания проводятся во время сдачи проекта разработчиком совместно с заказчиком. Все подсистемы испытываются одновременно на корректность взаимодействия подсистем, влияние подсистем друг на друга.

Во время приемочных испытаний оцениваются полнота и качество реализации функций, указанных в настоящем техническом задании.

Приемка результатов должна осуществляться в сроки, установленные таблицей 5.1. Во время приемочных испытаний оцениваются полнота и качество реализации функций, указанных в настоящем техническом задании.

### 6.2 Общие требования к приёмке работы

Приемка осуществляется представителями Заказчика и Исполнителя. Все создаваемые в рамках настоящей работы программные изделия передаются Заказчику.



## 7 Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие

Для обеспечения готовности объекта к вводу системы в действие следует провести следующий комплекс мероприятий:

- Загрузка файлов приложения;
- Загрузка базы данных;
- Связь базы данных и приложения;
- Проведение предварительных испытаний;
- Проверка приемочных испытаний.

## 8 Требования к документированию

Состав программной документации:

- Задание на курсовую работу;
- Техническое задание (ТЗ);
- Пояснительная записка (ПЗ);

## ОГЛАВЛЕНИЕ

1 ОБЗОР .....	21
1.1 Выбор предметной области .....	21
1.2 Выбор типа базы данных .....	21
1.3 Выбор средств разработки .....	21
2 ВОПРОСЫ ПРОЕКТИРОВАНИЯ БАЗЫ ДАННЫХ .....	23
2.1 Проектирование инфологической модели данных .....	23
2.2 Проектирование реляционной модели базы данных .....	26
2.3 Составление глоссария .....	28
2.4 Проектирование физической базы данных .....	33
2.5 Введение ролевого разграничения .....	37
3 ПРОЕКТИРОВАНИЕ ПРИЛОЖЕНИЯ БД .....	39
3.1 Windows Form .....	39
4 ОБЕСПЕЧЕНИЕ БЕЗОПАСНОСТИ ПЕРСОНАЛЬНЫХ ДАННЫХ .....	51
Заключение .....	58
Список использованных источников .....	58
Приложение А (обязательное) Листинг для главной формы .....	60
Приложение Б (обязательное) Листинг для формы с таблицей .....	62
Приложение В (обязательное) Листинг формы авторизации .....	67
Приложение Г (обязательное) Выдержка из ФЗ 152 .....	70
Приложение Д (обязательное) Выдержка из Приказа ФСТЭК 21 .....	73

## **Введение**

В процессе выполнения курсовой работы должна быть разработана автоматизированная информационная система, основанная на учебно-исследовательской базе данных, охватывающей сферу «Казино». Работа над курсовой работой будет проводиться в соответствии с этапами, определенными в задании.

В ходе выполнения проекта будут приобретены и усовершенствованы следующие навыки:

- Проектирование и создание баз данных, а также работа с ними.
- Разработка политики безопасности пользователей приложения.
- Написание SQL-запросов для взаимодействия с базой данных.
- Навыки программирования на языке C#.
- Создание проекта в среде разработки Visual Studio 2022.

# **1 ОБЗОР**

## **1.1 Выбор предметной области**

Предметная область, связанная работой казино, также характеризуется значительным объемом данных. В этом контексте активно используются базы данных и специализированные системы для обработки игровой информации. Задачи, связанные с анализом данных в игровой области, становятся особенно важными, поскольку важно иметь информацию о проводимых играх, для отслеживания возможных мошенников. Таким образом, в данной области также наблюдается высокая актуальность задач, связанных с обработкой и анализом данных.

## **1.2 Выбор типа базы данных**

Выбор типа базы данных представляет собой одно из ключевых решений, определяющих архитектуру проектирования базы данных и системы в целом. В данном случае было принято решение о предпочтении реляционных баз данных. Этот выбор основан на том, что реляционные базы данных обладают необходимыми характеристиками для поддержки функциональности, связанной с предметной областью работы. Кроме того, для данного типа баз данных доступно множество сред разработки, что упрощает процесс создания и поддержки системы. Так же реляционные СУБД популярны.

## **1.3 Выбор средств разработки**

Для разработки приложения, взаимодействующего с реляционной базой данных, имеется разнообразие платформ, методов и средств. В данном случае, в качестве языка программирования был выбран C#, представляющий собой объектно-ориентированный язык, способный создавать десктопные приложения.

В качестве фреймворка для разработки десктопных приложений был выбран Windows Forms, поддерживаемый Microsoft. Этот выбор был обоснован оценкой скорости разработки, удобством работы с базами данных, а также простотой реализации необходимых функциональных возможностей.

## 2 ВОПРОСЫ ПРОЕКТИРОВАНИЯ БАЗЫ ДАННЫХ

### 2.1 Проектирование инфологической модели данных

Для начала нужно неформально описать выбранную предметную область. Так как тема казино является достаточно обширной возьмем отдельную отрасль – отдельно взятую раздачу карт игрокам при какой-либо карточной игре. Данная предметная область будет являться покерным столом, за которым игроки взаимодействия с крупье будут играть. В данной предметной области будет возможность покупки фишек, раздачи карт, разыгрывание карт, а также получение результатов раздач. Покупка фишек будет осуществляться через кассира. В конце игры у каждого игрока будет изменяться баланс фишек, после этого игру можно будет начинать заново, либо же поменять стол или продать фишки обратно.

Далее представлен черный ящик, в который были внесены все входы и выходы процесса раздачи карт в конкретной раздаче (рисунок 2.1).

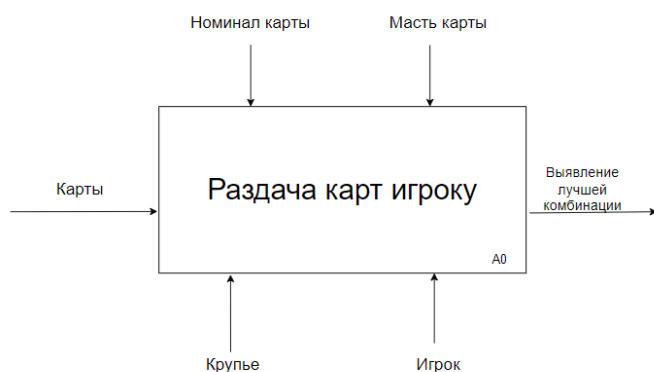


Рисунок 2.1 – Черный ящик

В данной модели есть пять блоков, в каждом из них прописаны входы, выходы, участники и ограничения для игры в покер. В первом блоке происходит перемешивание карт, в нём участвуют крупье и сами игроки, ограничением здесь выступает объем карточной колоды. Во втором блоке происходит раздача карт, здесь список участников остаётся неизменным, а в ограничения

добавляются номинал и масть карт. После чего в следующем блоке происходит принятие ставок, участники все те же, ограничением является бюджет фишек конкретного игрока. Последним блоком здесь будет выявление победителя в раздаче. Участники также неизменны, а ограничение в этом блоке заключается в лучшей комбинации карт за столом (Рисунок 2.2).

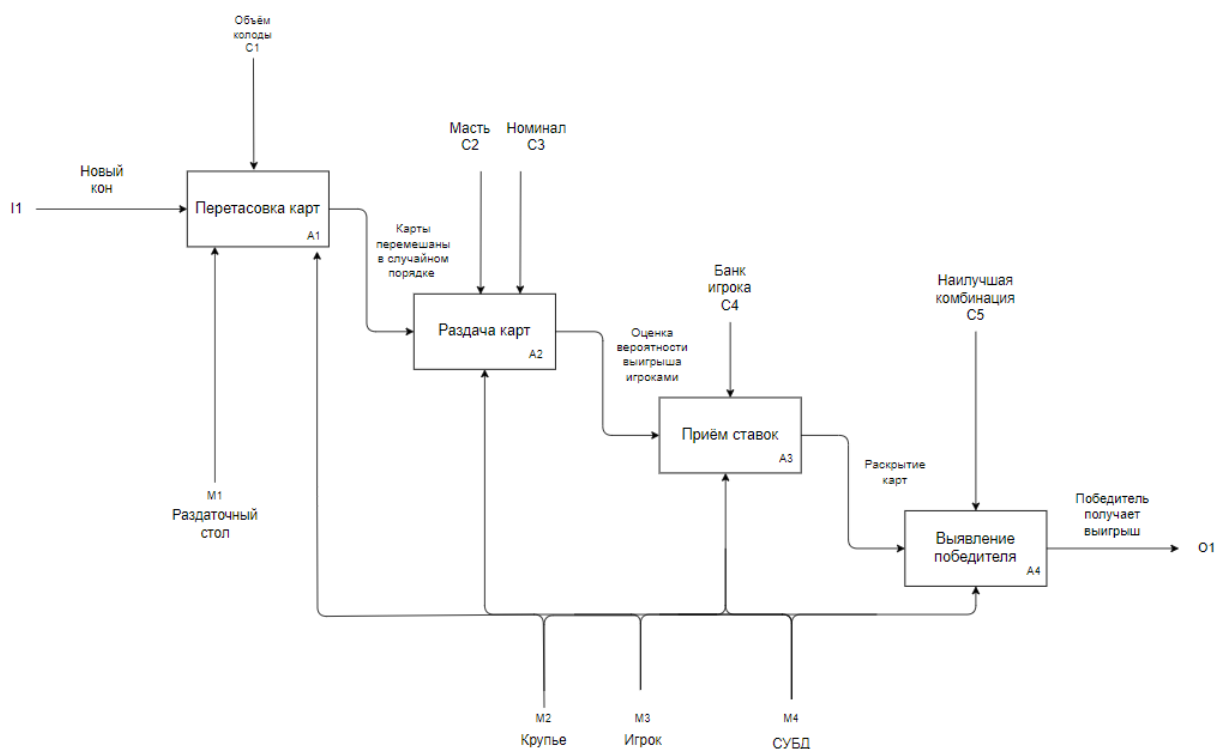


Рисунок 2.2 – IDEF0



На рисунке 2.3 представлена концептуальная модель ПО, где определены основные объекты ПО, связи между ними, а также их атрибуты.

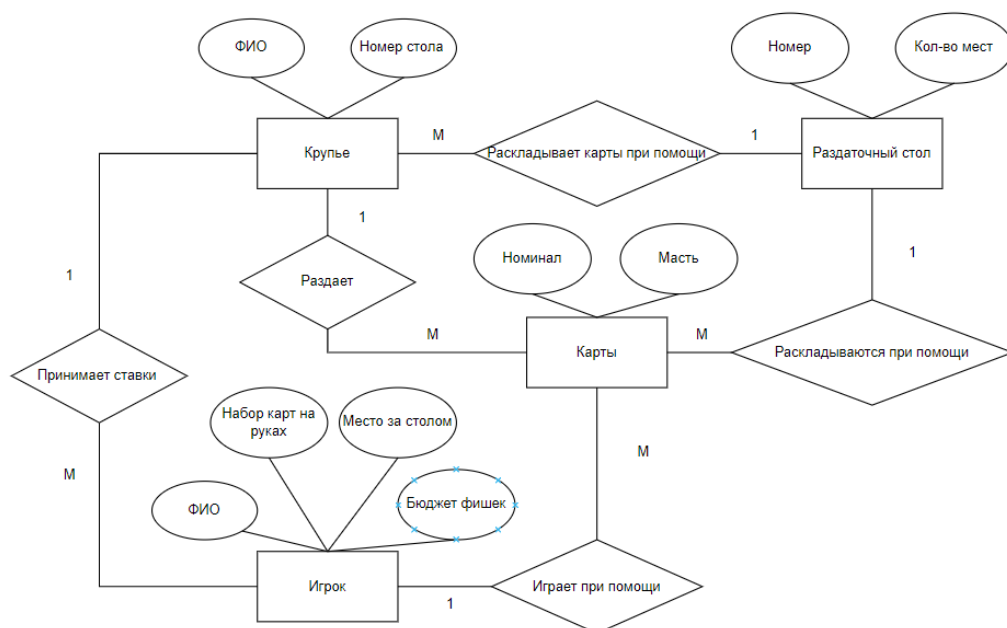


Рисунок 2.3 – Концептуальная модель ПО

На данной концептуальной модели представлены сущности и их атрибуты связи:

- Сущность Игрок с атрибутами ФИО, Набор карт на руках, Место за столом, Бюджет фишек игрока. Связь один ко многим с сущностью Карты;
- Сущность Крупье с атрибутами ФИО и Номер стола, за которым он работает. Связи один ко многим с сущностями Игрок и Карты;
- Сущность Раздаточный стол с атрибутами Номер стола, Количество мест за столом. Связь один ко многим с сущностью Карты.
- Сущность Карты с атрибутами Номинал и Масть.

## 2.2 Проектирование реляционной модели базы данных

Реляционная база данных – это тело связанной информации, сохраняемой в двумерных таблицах. Реляционная модель базы данных представлена на рисунке 2.4. В процессе нормализации базы данных были выполнены следующие действия: задействованы идентификационные номера для обеспечения уникальности, также были созданы дополнительные таблицы для выполнения третьей нормальной формы.

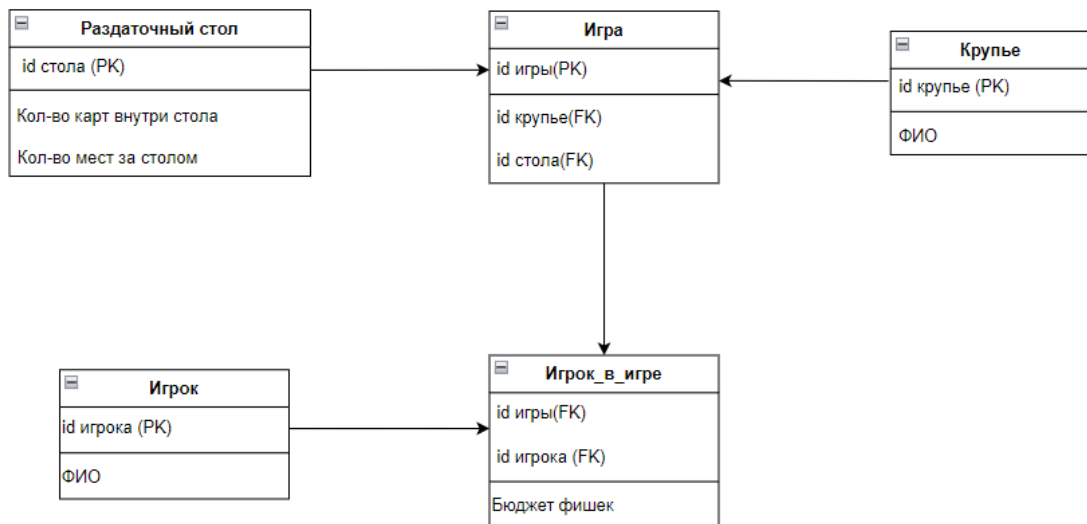


Рисунок 2.4 – Реляционная модель базы данных

Следующим этапом была составлена модель IDEF1X (рисунок 2.5).

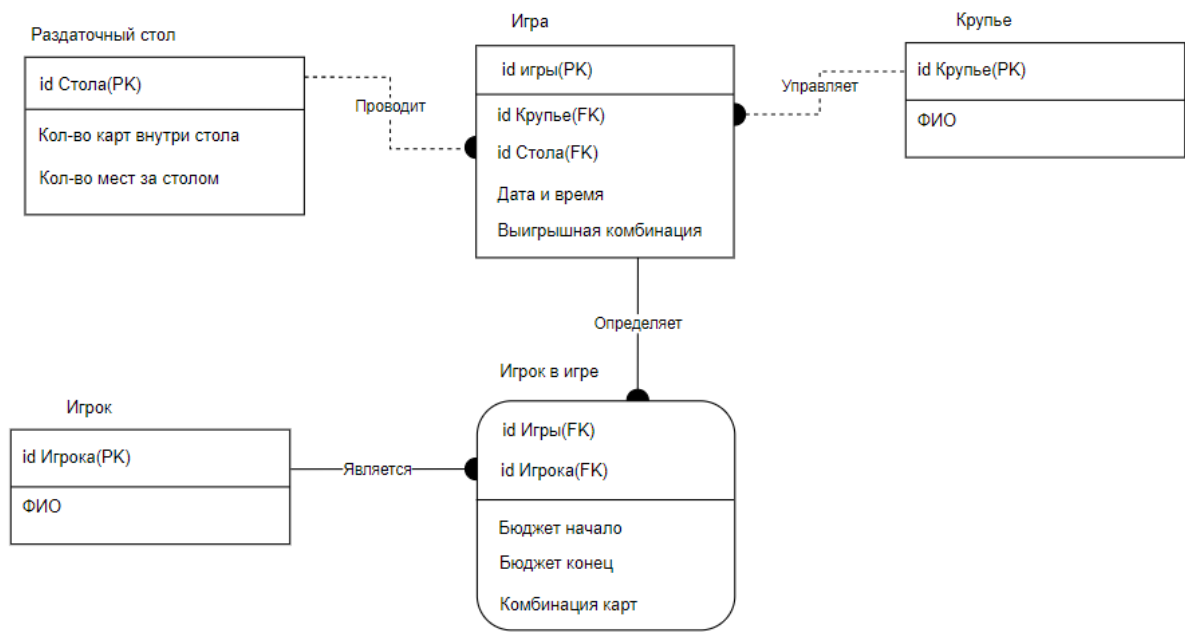


Рисунок 2.5 – IDEF1X

## 2.3 Составление глоссария

Далее для старта проектирования физической базы данных были созданы глоссарии, содержащие параметры для каждого из атрибутов таблиц.

В таблице 2.1 представлен глоссарий для таблицы Игрок, в которой содержится информация о зарегистрированных игроках. Здесь в качестве ограничений была выставлена длина атрибута ФИО и условие на автоинкрементирование первичного ключа id с шагом 1.

Таблица 2.1 – Игрок

Имя поля	Тип данных поля	Описание	Ограничения
Id Игрока	Счетчик	Уникальный номер игрока	Обязательное поле. Первичный ключ. Условие на значение: >0 Счетчик с 1 шаг 1.
ФИО	Текстовый	Идентифицирующие данные игрока	Обязательное поле. Размер поля 60.

В таблице 2.2 представлен глоссарий для таблицы Крупье. Далее был составлен глоссарий для таблицы Крупье, здесь так же, как и в таблице Игрок были прописаны ограничения на длину атрибутов.

Таблица 2.2 – Крупье

<b>Имя поля</b>	<b>Тип данных поля</b>	<b>Описание</b>	<b>Ограничения</b>
Id Крупье	Счетчик	Уникальный номер крупье	Обязательное поле. Первичный ключ. Условие на значение: >0. Счетчик с 1 шаг 1.
ФИО	Текстовый	Идентифицирующие данные крупье	Обязательное поле. Размер поля 60.

В таблице 2.3 представлен глоссарий для таблицы Игра. Далее был составлен глоссарий для таблицы Игра, здесь представлены ограничения на длину атрибута, обязательное поле, автоинкрементирование, а также ограничение на значение по умолчанию.

Таблица 2.3 – Игра

<b>Имя поля</b>	<b>Тип данных поля</b>	<b>Описание</b>	<b>Ограничения</b>
Id Игры	Счетчик	Уникальный номер игры	Обязательное поле. Первичный ключ. Условие на значение: >0. Счетчик с 1 шаг 1.
Id Крупье	Числовой	Уникальный номер крупье. Ссылается на таблицу «Крупье» поле «Id Крупье»	Обязательное поле. Внешний ключ.

Продолжение таблицы 2.3 – Игра

Id Стола	Числовой	Уникальный номер стола. Ссылается на таблицу «Раздаточный стол» поле «Id Стола».	Обязательное поле. Внешний ключ
Дата и Время	Временной	Дата и время проведения игры	Обязательное поле.
Выигрышная комбинация	Текстовый	Лучшая комбинация, выпавшая при раздаче в игре	Обязательное поле. Размер поля 50. По умолчанию значение «Ожидает результата»

В таблице 2.4 представлен глоссарий для таблицы Раздаточный стол. Здесь были применены такие ограничения как: максимальное и минимальное значение поля, ограничение на счётчик с шагом 1, а также ограничение на использование только целочисленных значений в атрибуте.

Таблица 2.4 – Раздаточный стол

Имя поля	Тип данных поля	Описание	Ограничения
Id Стола	Счетчик	Уникальный номер стола	Обязательное поле. Первичный ключ. Условие на значение: >0. Счетчик с 1 шаг 1.

Продолжение таблицы 2.4 – Раздаточный стол

Кол-во карт внутри стола	Числовой	Количество оставшихся карт внутри раздаточного стола	Обязательное поле. Условие на значение: $\geq 0$ . Максимальное число = 56. Целые числа.
Кол-во мест за столом	Числовой	Количество игровых мест, доступных для игры за столом	Обязательное поле. Условие на значение: $> 0$ . Целые числа.

В таблице 2.5 представлен глоссарий для таблицы Игрок в игре. Здесь были применены ограничение, относящиеся к внешним ключам таблицы, условие на минимальное значение поля, а также на значение по умолчанию.

Таблица 2.5 – Игрок в игре

Имя поля	Тип данных поля	Описание	Ограничения
Id Игры	Числовой	Уникальный номер игры. Ссылается на таблицу «Игра», поле «Id Игры»	Обязательное поле. Внешний ключ. Входит в состав Первичного ключа..
Id Игрока	Числовой	Уникальный номер игрока. Ссылается на таблицу «Игрок», поле «Id Игрока»	Обязательное поле. Внешний ключ. Входит в состав Первичного ключа.

Продолжение Таблицы 2.5 – Игрок в игре

Бюджет начало	Числовой	Имеющиеся на руках у игрока фишки до начала раздачи	Обязательное поле. Условие на значение: $>0$ . Целое число.
Бюджет конец	Числовой	Имеющиеся на руках у игрока фишки после окончания раздачи	Обязательное поле. Условие на значение: $\geq 0$ . Целое число.
Комбинация карт	Текстовый	Комбинация карт, с которой игрок провел раздачу	Обязательное поле. Название комбинации или «сброс». Размер поля 50.



## 2.4 Проектирование физической базы данных

Следующим этапом выполнения курсовой работы было проектирование физической базы данных по написанным в пункте 1.3 глоссариям. На рисунках

На рисунках 2.4 – 2.13 представлены проекты таблиц, а также первые пять записей в каждой из этих таблиц.

	Column Name	Data Type	Allow Nulls
🔑	id	int	<input type="checkbox"/>
	card_number	int	<input type="checkbox"/>
	places	int	<input type="checkbox"/>
▶			<input type="checkbox"/>

Рисунок 2.4 – Таблица Раздаточный стол

	id	card_num...	places
	1	10	5
	2	25	3
	3	42	7
	4	34	5
	5	54	6
▶*	NULL	NULL	NULL

Рисунок 2.5 – Первые пять записей в таблице Раздаточный стол

	Column Name	Data Type	Allow Nulls
🔑	player_id	int	<input type="checkbox"/>
	player_name	nvarchar(60)	<input type="checkbox"/>
▶			<input type="checkbox"/>

Рисунок 2.6 – Таблица Игрок

	player_id	player_name
	1	Иван Васильев
	2	Мария Птрова
	3	Иван Иванов
	5	Андрей Андреев
	6	Антон Антонов
▶*	NULL	NULL

Рисунок 2.7 – Первые пять записей в таблице Игрок

	Column Name	Data Type	Allow Nulls
🔑	croupier_id	int	<input type="checkbox"/>
	croupier_name	nvarchar(255)	<input checked="" type="checkbox"/>
▶			<input type="checkbox"/>

Рисунок 2.8 – Таблица Крупье

	croupier_id	croupier_name
	1	Елена Сергеева
	2	Дмитрий Васютин
	3	Сергей Селеванов
	4	Иван Иванов
	6	Иван Иванов
▶*	NULL	NULL

Рисунок 2.9 – Первые пять записей таблицы Крупье

	Column Name	Data Type	Allow Nulls
🔑	game_id	int	<input type="checkbox"/>
	croupier_id	int	<input type="checkbox"/>
	table_id	int	<input type="checkbox"/>
	date_time	smalldatetime	<input type="checkbox"/>
	best_combination	nvarchar(50)	<input type="checkbox"/>
▶			<input type="checkbox"/>

Рисунок 2.10 – Таблица Игра

	game_id	croupier_id	table_id	date_time	best_combi...
	1	1	2	2023-06-15 ...	Стрит флеш
	2	2	3	2023-06-15 ...	Ожидает р...
	3	1	2	2023-06-15 ...	Фулл хаус
	5	1	2	2024-01-23 ...	Фулл хаус
	7	2	3	2023-06-15 ...	Ожидает р...
►*	NULL	NULL	NULL	NULL	NULL

Рисунок 2.11 – Первые пять записей таблицы Игра

	Column Name	Data Type	Allow Nulls
🔑	player_id	int	<input type="checkbox"/>
🔑	game_id	int	<input type="checkbox"/>
	start_buget	int	<input type="checkbox"/>
	finish_buget	int	<input type="checkbox"/>
	card_combination	nvarchar(50)	<input type="checkbox"/>
►			<input type="checkbox"/>

Рисунок 2.12 – Таблица Игрок в игре

	player_id	game_id	start_buget	finish_buget	card_comb...
	1	2	700	900	Две пары
	2	3	1000	800	Сброс
	3	1	1200	1500	Фулл хаус
	3	2	1200	1500	Фулл хаус
	1	3	900	1800	Каре
►*	NULL	NULL	NULL	NULL	NULL

Рисунок 2.13 – Первые пять записей таблицы Игрок в игре

Так же по результатам физического проектирования при помощи встроенного инструмента SSMS была составлена диаграмма базы данных она представлена на рисунке 2.14.

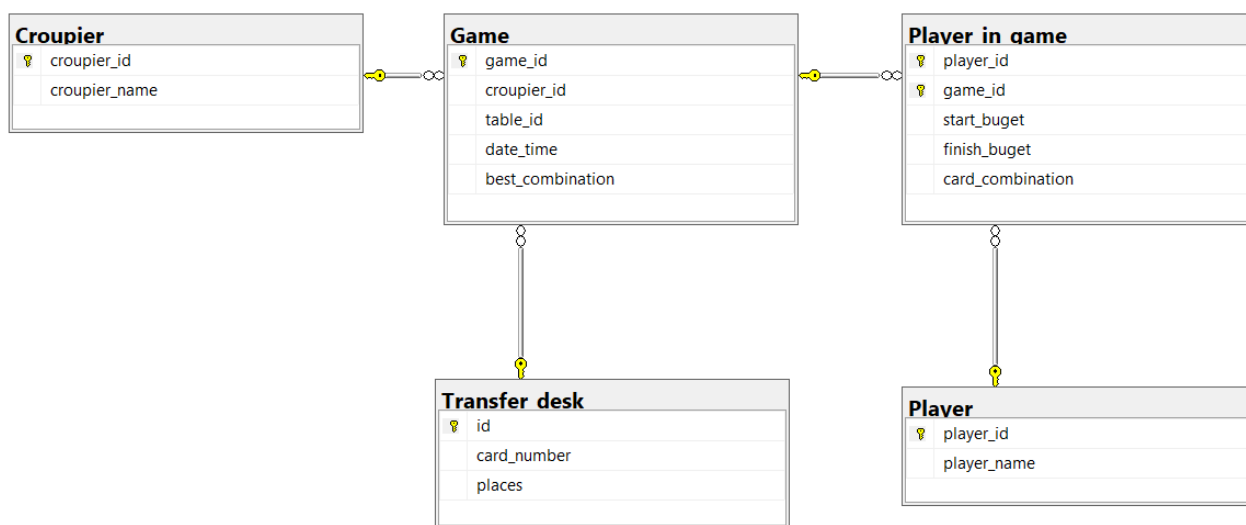


Рисунок 2.14 – Диаграмма базы данных

## 2.5 Введение ролевого разграничения

Далее была разработана матрица доступа (Таблица 2.6), в которую были занесены все таблицы, а также роли пользователей базы данных.

Таблица 2.6 – Матрица доступа

Пользователи Таблицы	Администратор	Кассир	Крупье	Игрок
Croupier	SELECT, INSERT, UPDATE, DELETE	SELECT	SELECT	-
Game	SELECT, INSERT, UPDATE, DELETE	SELECT	SELECT, INSERT, UPDATE	SELECT
Player	SELECT, INSERT, UPDATE, DELETE	SELECT, INSERT, UPDATE	-	-
Player_in_game	SELECT, INSERT, UPDATE, DELETE	SELECT	SELECT, INSERT, UPDATE	-
Transfer_desk	SELECT, INSERT, UPDATE, DELETE	SELECT	SELECT, INSERT, UPDATE	-

В данной курсовой работе необходима реализация двух ролей, определённых матрицей доступа. Для реализации были выбраны роли администратора и крупье. Разграничение доступа было выполнено программно при помощи специально созданной таблицы в базе данных (рисунок 2.15). Данная таблица состоит из логинов и паролей зарегистрированных пользователей, их персональных идентификаторов, а также из идентификатора роли конкретного пользователя. Здесь идентификатор 1 означает роль администратора, а 0 – Крупье.

	id_user	login_user	password_user	is_admin
1	1	admin	admin	1
2	2	kirill	111	1
3	3	administrator	123	0
4	4	123	123	0

Рисунок 2.15 – Таблица зарегистрированных пользователей

После этого в проекте десктопного приложения был создан специальный класс, который отвечает за разграничение доступа в программе. При помощи данного класса были ограничены возможности для пользователей, не имеющих доступа к конкретным данным, а также замаскированы некоторые атрибуты таблиц (рисунок 2.16).

```

1  using System;
2  using System.Collections.Generic;
3  using System.Data.SqlClient;
4  using System.Linq;
5  using System.Text;
6  using System.Threading.Tasks;
7  using System.Xml.Linq;
8
9  namespace Course_Work
10 {
11     Ссылка 12
12     public class CheckUser
13     {
14         Ссылка 1
15         DataBase _dataBase;
16
17         Ссылка 12
18         public string Login { get; set; }
19
20         Ссылка 12
21         public bool IsAdmin { get; }
22
23         Ссылка 0
24         public string Status => IsAdmin ? "Admin" : "Croupier";
25
26         Ссылка 1
27         public CheckUser(string login, bool is_admin)
28         {
29             Login = login.Trim();
30             IsAdmin = is_admin;
31         }
32     }
33 }

```

Рисунок 2.16 – Класс «Check user»

### 3 ПРОЕКТИРОВАНИЕ ПРИЛОЖЕНИЯ БД

#### 3.1 Windows Form

Следующим этапом выполнения была работа в Windows Form. Для этого зайдём в Visual Studio 2022 и создадим новый проект на .NET Framework 4.7.2 (рисунок 3.1).

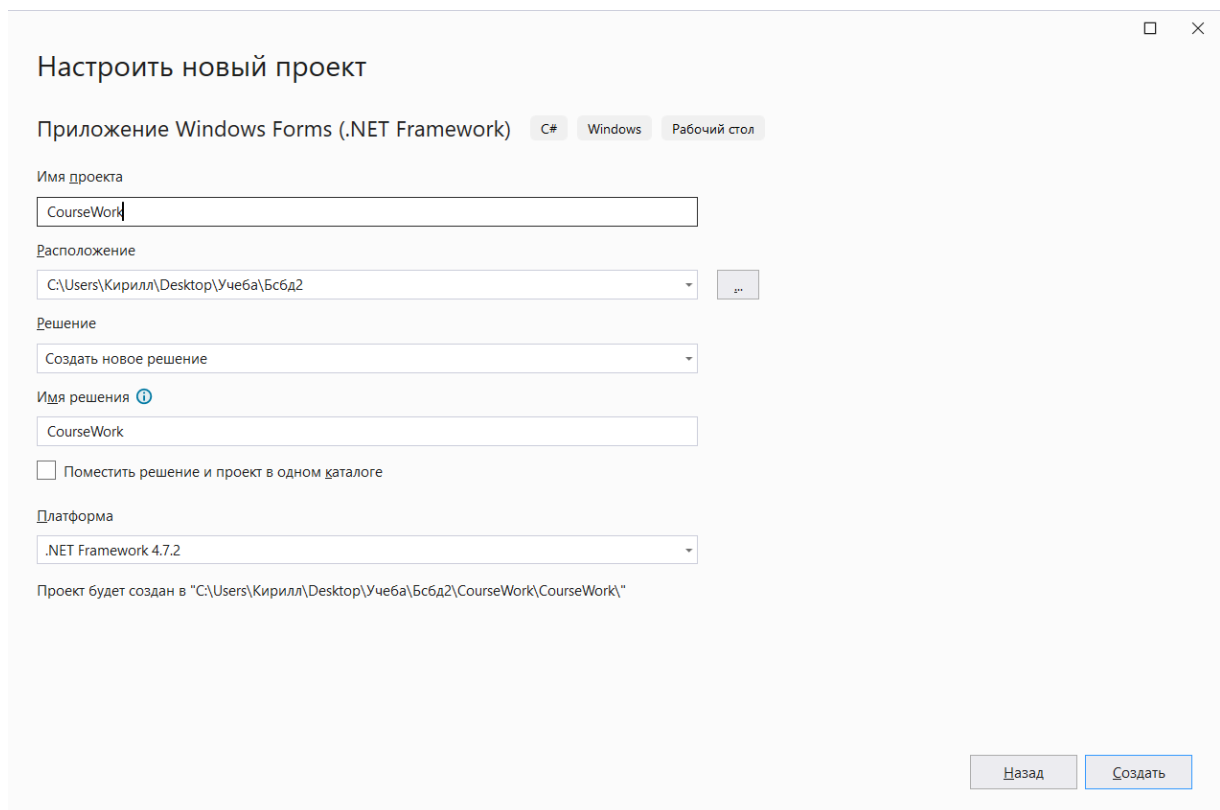


Рисунок 3.1 – Создание нового проекта

Первым делом был создан специальный класс, который отвечает за подключение к базе данных. Для работы с базами данных SQL подключается специальная библиотека, после чего создаётся экземпляр класса этой библиотеки, в который в качестве параметра передаётся строка подключения. Класс подключения базы данных содержит 2 метода по открытию и закрытию подключения к базе (рисунок 3.2).

```

1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6  using System.Data.SqlClient;
7
8  namespace Course_Work
9  {
10     internal class DataBase
11     {
12
13         Ссылка: 15
14         SqlConnection sqlConnection = new SqlConnection
15             ("Data Source = DESKTOP-DB318D2; Initial Catalog = CASINOBASE; Integrated Security = True");
16
17         Ссылка: 21
18         public void openConnection()
19         {
20             if (sqlConnection.State == System.Data.ConnectionState.Closed)
21             {
22                 sqlConnection.Open();
23             }
24         }
25
26         Ссылка: 11
27         public void closeConnection()
28         {
29             if (sqlConnection.State == System.Data.ConnectionState.Open)
30             {
31                 sqlConnection.Close();
32             }
33         }
34
35         Ссылка: 28
36         public SqlConnection getConnection()
37         {
38             return sqlConnection;
39         }
40     }
41 }

```

Рисунок 3.2 – Класс DataBase для подключения базы данных к проекту

После этого была создана форма авторизации. Данная форма содержит 2 текстовых поля, в которые вносятся логин и пароль. После нажатия кнопки «Войти» отправляется запрос в базу данных на проверку соответствия введённых данных одной из строчек таблицы, содержащей список зарегистрированных пользователей. Также форма содержит кнопку перехода на страницу регистрации, которая будет реализована следующей после авторизации (рисунок 3.3).



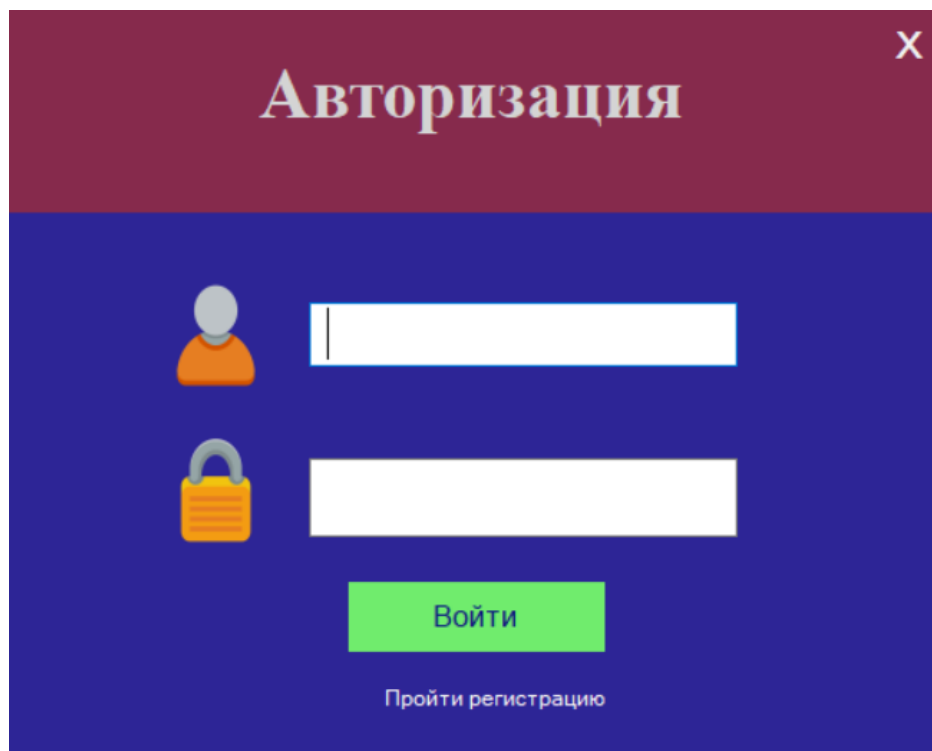


Рисунок 3.3 – Форма авторизации

Функция, отвечающая за работу кнопки «Войти» представлена на рисунке 3.4. в функционале данной кнопки заложено сравнение введенных данных с существующими записями. Если соответствующая запись существует, происходит переход к основной форме, если нет, отображается сообщение об ошибке.

```
private void button1_Click(object sender, EventArgs e)
{
    var loginUser = loginBox.Text;
    var passUser = passwordBox.Text;

    SqlDataAdapter adapter = new SqlDataAdapter();
    DataTable dt = new DataTable();

    string querystring = $"select id_user, login_user, password_user, is_admin from register where " +
        $"login_user = '{loginUser}' and password_user = '{passUser}'";

    SqlCommand command = new SqlCommand(querystring, data.getConnection());

    adapter.SelectCommand = command;
    adapter.Fill(dt);

    if (dt.Rows.Count == 1)
    {
        var user = new CheckUser(dt.Rows[0].ItemArray[1].ToString(), Convert.ToBoolean(dt.Rows[0].ItemArray[3]));
        MessageBox.Show("Вы успешно вошли!", "Успешно!", MessageBoxButtons.OK, MessageBoxIcon.Information);
        MainForm frm1 = new MainForm(user);
        this.Hide();
        frm1.ShowDialog();
        this.Show();
    }
    else MessageBox.Show("Такого аккаунта не существует!", "Аккаунта не существует!", MessageBoxButtons.OK, MessageBoxIcon.Warning);
}
```

Рисунок 3.4 – Функция кнопки «Войти»

Функция лейбла для перехода к странице регистрации представлена на рисунке 3.5.

```
private void reg_button_Click(object sender, EventArgs e)
{
    reg_button.ForeColor = Color.Gray;

    RegForm frm2 = new RegForm();
    this.Hide();
    frm2.ShowDialog();
    this.Show();
}
```

Рисунок 3.5 – Функция перехода к форме регистрации

Следующим этапом стало создание страницы регистрации. Она имеет схожие со страницей авторизации поля для ввода логина и пароля, а также выпадающее меню с выбором роли пользователя, от которой зависит уровень доступа к приложению. При нажатии на кнопку «Зарегистрироваться» базе данных отправляется запрос на добавление пользователя. Если пользователь под таким логином уже существует, система выдаст предупреждение об этом (рисунок 3.6).



Рисунок 3.6 – Страница регистрации

Функционал кнопки «Зарегистрироваться» объединён в специальный метод (рисунок 3.7). Проверка наличия повторяющихся строчек таблицы также производится при помощи специальной функции (рисунок 3.8).

```
private void buttonReg_Click(Object sender, EventArgs e)
{
    var login = loginBox2.Text;
    var password = passwordBox2.Text;
    var index = comboBox1.SelectedIndex;

    string querystr = $"insert into register(login_user, password_user, is_admin) values('{login}', '{password}', {index})";
    SqlCommand cmd = new SqlCommand(querystr, dataBase.getConnection());

    dataBase.openConnection();

    if (checkuser())
    {
        return;
    }
    if (cmd.ExecuteNonQuery() == 1)
    {
        MessageBox.Show("Аккаунт успешно создан!", "Успех!");
        this.Close();
    }
    else
    {
        MessageBox.Show("Аккаунт не создан!");
    }
    dataBase.closeConnection();
}
```

Рисунок 3.7 – Функция кнопки «Зарегистрироваться»

```
private Boolean checkuser()
{
    var logUser = loginBox2.Text;
    var passUser = passwordBox2.Text;

    SqlDataAdapter adapter = new SqlDataAdapter();
    DataTable dt = new DataTable();
    string qstring = $"select id_user, login_user, password_user, is_admin from register where login_user = '{logUser}'";

    SqlCommand command = new SqlCommand(qstring, dataBase.getConnection());

    adapter.SelectCommand = command;
    adapter.Fill(dt);

    if (dt.Rows.Count > 0)
    {
        MessageBox.Show("Такой логин уже используется!");
        return true;
    }
    else return false;
}
```

Рисунок 3.8 – Функция проверки наличия существующих аккаунтов

При открытии новой формы, активная скрывается. При нажатии на кнопку закрытия формы, пользователь возвращается на родительскую форму.

После прохождения регистрации, пользователь может попасть на основную форму приложения, которая содержит список из всех существующих таблиц в базе данных и доступных конкретному пользователю в зависимости от роли. Так различия между ролями Администратор и Крупье заключаются в том, что второму недоступна таблица «Игрок» (рисунки 3.9, 3.10).

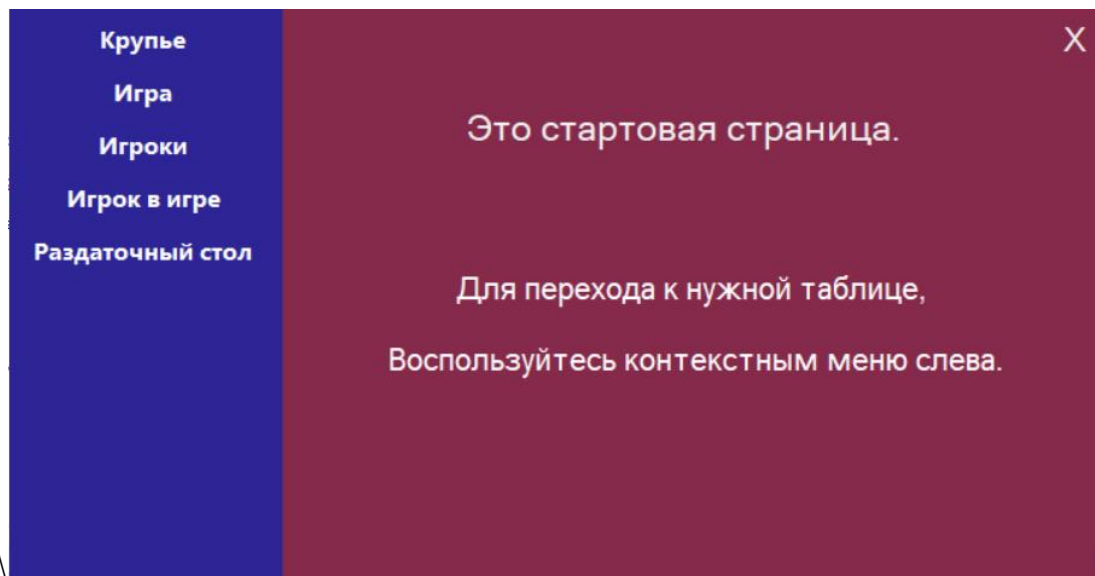


Рисунок 3.9 – Основная форма приложения от лица Администратора

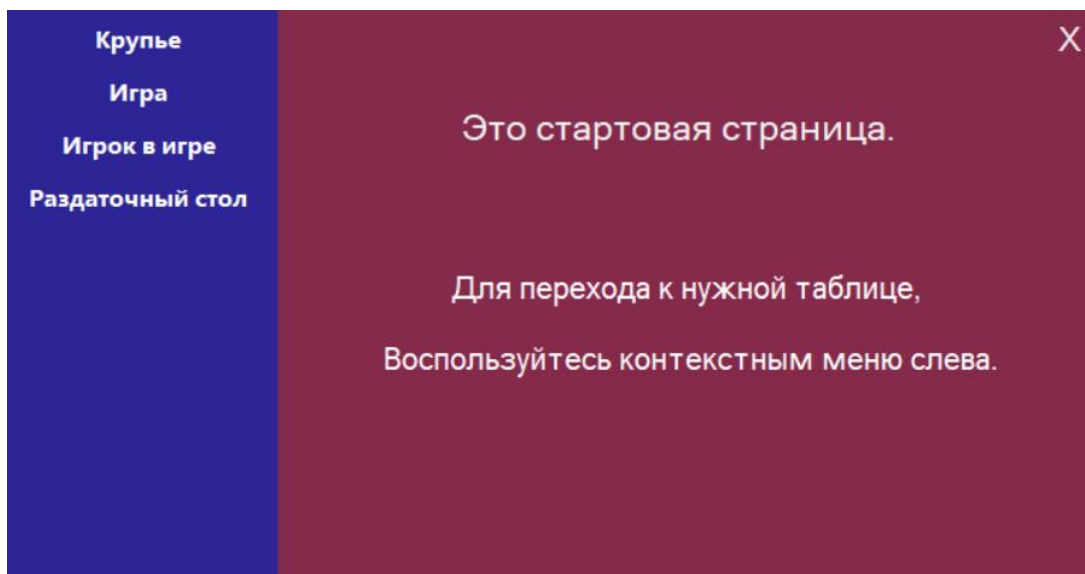


Рисунок 3.10 - Основная форма приложения от лица Крупье

Это реализовано при помощи ранее описанного класса «CheckUser». В

переменную записывается значение индекса роли авторизованного пользователя, и если оно соответствует роли Крупье, то таблица «Игрок» скрывается из списка (рисунок 3.11).

```
public MainForm(CheckUser user)
{
    InitializeComponent();

    StartPosition = FormStartPosition.CenterScreen;

    _user = user;

    thirdButton.Visible = _user.IsAdmin;
}
```

Рисунок 3.11 – Разграничение доступа к таблице «Игрок»

Каждая кнопка в боковом меню отправляет к таблице базы данных с соответствующим названием. На рисунке 3.12 представлен пример формы, на которые ссылаются кнопки в боковом меню.

	id игры	id крупье	id стола	дата игры	лучшая комбинация
▶	1	1	2	15.06.2023 13:2...	Стрит флеш
	2	2	3	15.06.2023 13:2...	Ожидает резуль...
	3	1	2	15.06.2023 13:2...	Фулл хаус
	5	1	2	23.01.2024 13:2...	Фулл хаус
	7	2	3	15.06.2023 13:2...	Ожидает резуль...

id Игры

id Крупье

id Стола

Дата игры

Лучш. комб.

Поиск

Новая запись

Изменить

Удалить

Рисунок 3.12 – Форма Таблицы «Игра»

Основным компонентом формы является Data Grid View. В нём происходит вывод таблиц на форме. Для заполнения данными этого компонента были созданы экземпляры классов «DataBase» и «CheckUser», который был использован для шифрования данных.

В первую очередь в Data Grid View было создано необходимое количество столбцов (рисунок 3.13). Помимо основных столбцов, был создан дополнительный столбец «IsNew», который будет отвечать за состояние каждой записи в таблице. Из возможных значений в данном столбце присутствуют «Existed», «New», «Modified», «Deleted», и значение по умолчанию «Modified New»

```
private void CreateColumns()
{
    dataGridView1.Columns.Add("game_id", "id игры");
    dataGridView1.Columns.Add("croupier_id", "id крупье");
    dataGridView1.Columns.Add("table_id", "id стола");
    dataGridView1.Columns.Add("date_time", "дата игры");
    dataGridView1.Columns.Add("best_combination", "лучшая комбинация");
    dataGridView1.Columns.Add("IsNew", "ColumnName");

    dataGridView1.Columns["IsNew"].Visible = false;
}

Ссылка: 2
private void ReadSingleRow(DataGridView dgw, IDataRecord record)
{
    dgw.Rows.Add(record.GetInt32(0),
        record.GetInt32(1),
        record.GetInt32(2),
        record.GetDateTime(3),
        record.GetString(4),
        Rowstate.ModifiedNew);
}
```

Рисунок 3.13 – Листинг программы по созданию таблиц в Data Grid View

Далее были созданы методы по обновлению данных, записанных в Data Grid View и отображению формы таблицы (рисунок 3.14). Метод обновления данных перезаписывает данные до тех пор, пока не дойдёт до последней строки таблицы. В методе по выгрузке данных реализована шифровка данных в конкретных столбцах, при условии, что пользователь не имеет к ним доступа.

```

private void RefreshDataGrid(DataGridView dgw)
{
    dgw.Rows.Clear();

    string queryString = $"select * from Game";

    SqlCommand cmd = new SqlCommand(queryString, _dataBase.getConnection());

    _dataBase.openConnection();

    SqlDataReader reader = cmd.ExecuteReader();

    while (reader.Read())
    {
        ReadSingleRow(dgw, reader);
    }
    reader.Close();
}

Ссылка: 1
private void Game_Load(object sender, EventArgs e)
{
    CreateColumns();
    RefreshDataGrid(dataGridView1);

    if (!_user.IsAdmin)
    {
        for (int i = 0; i < dataGridView1.Rows.Count; i++)
        {
            DataGridViewRow row = dataGridView1.Rows[i];
            row.Cells[3].Value = "#####";
        }
    }
}

```

Рисунок 3.14 – Методы по заполнению и обновлению данных в Data Grid View

Следующим шагом стала реализация вывода данных в соответствующие столбцам таблицы текстовые поля. Это происходит при помощи привязки строк к соответствующим им индексам (рисунок 3.15).

```

private void dataGridView1_CellClick(object sender, DataGridViewCellEventArgs e)
{
    selectedRow = e.RowIndex;

    if (e.RowIndex >= 0)
    {
        DataGridViewRow row = dataGridView1.Rows[selectedRow];

        game_id_tb.Text = row.Cells[0].Value.ToString();
        croupier_id_tb.Text = row.Cells[1].Value.ToString();
        table_id_tb.Text = row.Cells[2].Value.ToString();
        date_tb.Text = row.Cells[3].Value.ToString();
        combo_tb.Text = row.Cells[4].Value.ToString();
    }
}

```

Рисунок 3.15 – Метод по выводу данных в соответствующие поля

После этого на форме был реализован функционал каждой кнопки. Метод по обновлению данных был привязан к нажатию кнопки «Обновить страницу» в верхней части формы. Для кнопки «Новая запись» был создан специальный метод. Логика его работы такова, что пользователь заносит в таблицу новые данные при помощи соответствующих полей. Если при создании новой записи не вызывается исключений, запись вносится в таблицу. Если же исключение вызывается, запись не заносится в таблицу и всплывает соответствующее окно с предупреждением (рисунок 3.16).

```
private void button1_Click(object sender, EventArgs e)
{
    _dataBase.openConnection();

    var idCroupier = croupier_id_tb.Text;
    var idTable = table_id_tb.Text;
    var date = date_tb.Text;
    var best_comb = combo_tb.Text;

    if (date != "")
    {
        try
        {
            var addQuery = $"insert into Game (croupier_id, table_id, date_time, best_combination) values ('{idCroupier}', '{idTable}', '{date}', '{best_comb}')";
            var command = new SqlCommand(addQuery, _dataBase.getConnection());

            command.ExecuteNonQuery();

            MessageBox.Show("Строка добавлена в таблицу!", "Успех!", MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
        catch (Exception)
        {
            MessageBox.Show("Проверьте правильность заполнения полей!", "Ошибка!", MessageBoxButtons.OK, MessageBoxIcon.Information);
            return;
        }
    }
    else { MessageBox.Show("Ошибка заполнения таблицы!", "Ошибка!", MessageBoxButtons.OK, MessageBoxIcon.Information); }
    _dataBase.closeConnection();
}
```

Рисунок 3.16 – Метод по созданию новой записи в таблице

Для кнопки удаления записи был реализован функционал по стиранию записи из таблицы в соответствие с индексом выделенной строки. Логика работы такова, что при нажатии на кнопку удаления, значение в последнем столбце таблицы соответствующей записи меняется на «Deleted». После этого происходит обновление данных в таблице. Строка скрывается из Data Grid View, а базе данных отправляется запрос на удаление строчки из таблицы (рисунок 3.17).



```

private void deleteRow()
{
    try
    {
        int index = dataGridView1.CurrentCell.RowIndex;

        dataGridView1.Rows[index].Visible = false;

        if (dataGridView1.Rows[index].Cells[0].Value.ToString() == string.Empty)
        {
            dataGridView1.Rows[index].Cells[5].Value = Rowstate.Deleted;
            return;
        }
        dataGridView1.Rows[index].Cells[5].Value = Rowstate.Deleted;
    }
    catch (Exception) { MessageBox.Show("Ничего не выбрано!", "Ошибка!", MessageBoxButtons.OK, MessageBoxIcon.Information); }
}

```

Рисунок 3.17 – Метод по удалению строки из таблицы

Для кнопки «Изменить» была реализована логика по изменению данных в записи по выбранному индексу. После нажатия происходит проверка на корректность введённых данных. Если возникает исключение, появляется окно с предупреждением, если нет, изменения вносятся в таблицу.

```

private void Change()
{
    try
    {
        var selectedRowIndex = dataGridView1.CurrentCell.RowIndex;

        var IDGame = game_id_tb.Text;
        var IDCroupier = croupier_id_tb.Text;
        var IDTable = table_id_tb.Text;
        var DateTime = date_tb.Text;
        var combination = combo_tb.Text;

        if (dataGridView1.Rows[selectedRowIndex].Cells[0].Value.ToString() != string.Empty)
        {
            dataGridView1.Rows[selectedRowIndex].SetValues(IDGame, IDCroupier, IDTable, DateTime, combination);
            dataGridView1.Rows[selectedRowIndex].Cells[5].Value = Rowstate.Modified;
        }
        else { MessageBox.Show("Ошибка заполнения таблицы!", "Ошибка!", MessageBoxButtons.OK, MessageBoxIcon.Information); }
    }
    catch (Exception) { MessageBox.Show("Ничего не выбрано!", "Ошибка!", MessageBoxButtons.OK, MessageBoxIcon.Information); }
}

```

Рисунок 3.18 – Метод по изменению данных

И в последнюю очередь был реализован метод по применению всех вносимых изменений. В нём в зависимости от значения из последнего столбца таблицы базе данных посылаются соответствующие запросы к базе данных и происходит обновление Data Grid View для того, чтобы применённые изменения вступали в силу сразу же (рисунок 3.19).

```

new private void Update()
{
    _dataBase.openConnection();

    for (int index = 0; index < dataGridView1.Rows.Count; index++)
    {
        var rowState = (Rowstate)dataGridView1.Rows[index].Cells[5].Value;

        if (rowState == Rowstate.Existed) continue;

        if (rowState == Rowstate.Deleted)
        {
            try
            {
                var id = Convert.ToInt32(dataGridView1.Rows[index].Cells[0].Value);
                var deleteQuery = $"delete from Game where game_id = {id}";

                var command = new SqlCommand(deleteQuery, _dataBase.getConnection());
                command.ExecuteNonQuery();
            }
            catch (Exception)
            {
                MessageBox.Show("На данный id ссылается другая таблица!", "Ошибка!", MessageBoxButtons.OK, MessageBoxIcon.Information);
                RefreshDataGrid(dataGridView1);
            }
        }

        if (rowState == Rowstate.Modified)
        {
            var id_game = dataGridView1.Rows[index].Cells[0].Value.ToString();
            var id_croupier = dataGridView1.Rows[index].Cells[1].Value.ToString();
            var id_table = dataGridView1.Rows[index].Cells[2].Value.ToString();
            var date = dataGridView1.Rows[index].Cells[3].Value.ToString();
            var combo = dataGridView1.Rows[index].Cells[4].Value.ToString();

            var changeQuery = $"update Game set croupier_id = '{id_croupier}', table_id = '{id_table}', " +
                $"date_time = '{date}', best_combination = '{combo}' where game_id = '{id_game}'";

            var command = new SqlCommand(changeQuery, _dataBase.getConnection());
            command.ExecuteNonQuery();
        }
    }

    _dataBase.closeConnection();

    if (!_user.IsAdmin)
    {
        for (int i = 0; i < dataGridView1.Rows.Count; i++)
        {
            DataGridViewRow row = dataGridView1.Rows[i];
            row.Cells[3].Value = "*****";
        }
    }
}

```

Рисунок 3.19 – Метод по применению изменений

Аналогично были созданы для всех остальных таблиц базы данных с учётом разграничения доступа к данным, ограничений атрибутов таблиц и типов данных этих атрибутов.

Полные листинги функций и методов, описанных при создании приложения, находятся в приложениях к пояснительной записке. Итоговая версия проекта, а также весь код доступны по ссылке на репозиторий [https://github.com/GreateK/Course\\_work](https://github.com/GreateK/Course_work).

## 4 ОБЕСПЕЧЕНИЕ БЕЗОПАСНОСТИ ПЕРСОНАЛЬНЫХ ДАННЫХ

В качестве системы, для которой будет составлена модель угроз, будет использоваться логика работы казино. Демонстрационный рисунок системы представлен на рисунке 1.1.

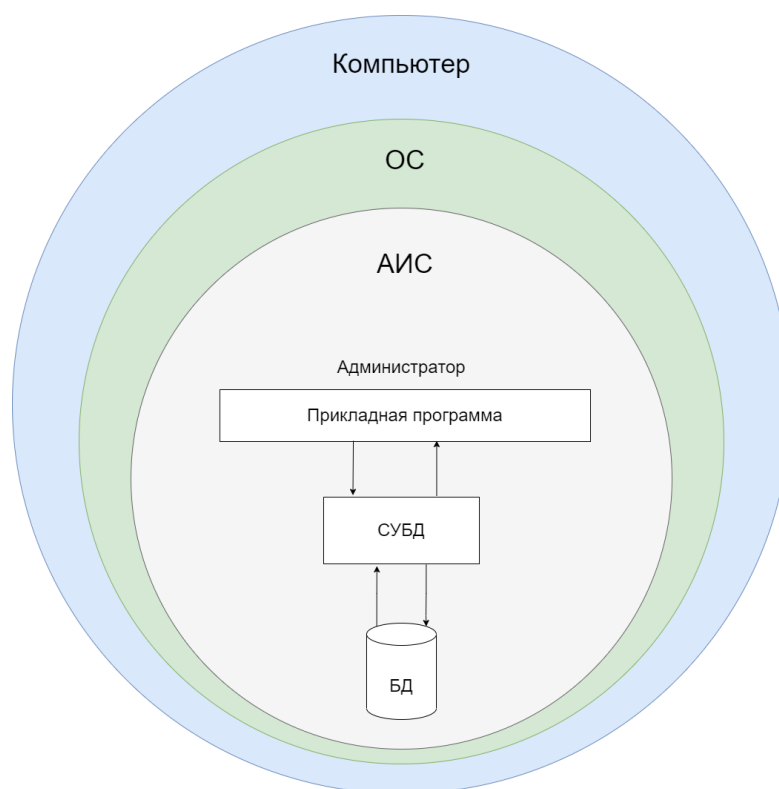


Рисунок 1.1 – Система работы казино

Как видно из рисунка 1.1, инвентаризация АИС состоит из:

- 1) Компьютер;
- 2) Операционная система;
- 3) Прикладная программа;
- 4) СУБД (Система управления базами данных);
- 5) База данных.

Далее, стоит уточнить, что операционной системой является «Windows 10». Компьютер с «Windows 10» в роли сервера. В качестве СУБД будет

использована программа – «SQL Server Management Studio». Прикладные программы будут являться программным обеспечением для сотрудников предприятия. Программы будут написаны на языки программирования «С#» с помощью фреймворков «.Net Framework», а также «ADO.NET». В качестве сервера будет являться «SQL SERVER».

В базе данных содержатся общие персональные данные. Согласно статье 19 федерального закона 152 ФЗ представленному в Приложении Г. Уровни защищенности представлены в Постановлении Правительства №1119. В 13 пункте сказано, что для обеспечения 4-го уровня защищенности персональных данных при их обработке в информационных системах необходимо выполнение следующих требований:

а) организация режима обеспечения безопасности помещений, в которых размещена информационная система, препятствующего возможности неконтролируемого проникновения или пребывания в этих помещениях лиц, не имеющих права доступа в эти помещения;

б) обеспечение сохранности носителей персональных данных;

в) утверждение руководителем оператора документа, определяющего перечень лиц, доступ которых к персональным данным, обрабатываемым в информационной системе, необходим для выполнения ими служебных (трудовых) обязанностей;

г) использование средств защиты информации, прошедших процедуру оценки соответствия требованиям законодательства Российской Федерации в области обеспечения безопасности информации, в случае, когда применение таких средств необходимо для нейтрализации актуальных угроз.

Полный перечень состава и содержания мер по обеспечению безопасности персональных данных, необходимых для обеспечения каждого из уровней защищенности персональных данных представлен в Приложении Д.

В данной курсовой работе были предусмотрены следующие технические меры:

- Идентификация и аутентификация пользователей, являющихся работниками оператора,

- Реализация необходимых методов (дискреционный, мандатный, ролевой или иной метод), типов (чтение, запись, выполнение или иной тип) и правил разграничения доступа;

- И др.

Так же были разработаны следующие организационно правовые меры:

- Составление соглашения о сборе персональных данных (Приложение Е);

Далее, было определено недопустимое событие: несанкционированное изменение данных в базе (целостность).

Для построения модели угроз необходимо обратиться в банк данных угроз «ФСТЭК», которая расположена на [официальном сайте](#). На складскую систему могут распространяться следующие угрозы, которые перечислены в [базе данных угроз ФСТЭКа](#): УБИ: 6, 7, 8, 12, 15, 18, 23, 25, 27, 28, 29, 36, 50, 51, 60, 61, 63, 67, 68, 74, 83, 88, 89, 91, 95, 102, 114, 115, 116, 122, 124, 125, 126, 127, 130, 139, 140, 143, 145, 146, 152, 157, 158, 179, 198, 207.

Для выбора мер защиты будут использованы следующие УБИ.

- 1) УБИ 6: Угроза внедрения кода или данных;
- 2) УБИ 15: Угроза доступа к защищаемым файлам с использованием обходного пути;
- 3) УБИ 102: Угроза опосредованного управления группой программ через совместно используемые данные;
- 4) УБИ 158: Угроза форматирования носителей информации;

Далее, в таблице 1 будут продемонстрированы угрозы, их реализации, а также меры защиты. Меры защиты были найдены в [приказе ФСТЭК России от 11 февраля 2013 г. N 17](#).

Таблица 1 – Анализ угроз, их реализаций и мер защиты

УБИ	Реализация УБИ	Меры защиты	Классы защищенности		
			3	2	1
6	<u>СП.4.2 Внедрение вредоносного программного обеспечения через съемные носители информации</u>	<u>АВЗ.1 Реализация антивирусной защиты</u>	+	+	+
		<u>ЗНИ.5.3 Контроль доступа пользователей к разрешенным к использованию интерфейсам ввода (вывода)</u>		+	+
15	<u>СП.2.10 Использование недостатков конфигурации сетевого оборудования, связанных с отсутствием или некорректной фильтрацией трафика</u>	<u>ЗИС.35.1 Фильтрация сетевого трафика, в том числе между внешними сетями и внутренними, в том числе при организации сетевого обмена с сетями связи общего пользования</u>	+	+	+
102	<u>СП.8.6 Обнаружение доступных общих сетевых каталогов</u>	<u>АУД.5.1 Применение систем анализа сетевого трафика, предназначенных для перехвата потоков данных и обнаружения признаков сложных, чаще всего целевых атак</u>	+	+	+
158	<u>Получение несанкционированного доступа к хранилищу данных</u>	<u>ДНС.3.1 Создание альтернативных мест хранения и обработки информации на случай возникновения нештатных ситуаций</u>			+

Следующим шагом будут описаны тактики и основные техники из методического документа от 5 февраля 2021 г ФСТЭК в таблице 2.

Таблица 2 - Перечень основных тактик и соответствующих им типовых техник, используемых для построения сценариев реализации угроз безопасности информации

№	Тактика	Основные техники
ТЗ	Внедрение и исполнение вредоносного программного обеспечения в системах и сетях Тактическая задача: получив доступ к узлу сети или системы, нарушитель стремится внедрить в его	ТЗ.1. Автоматический запуск скриптов и исполняемых файлов в системе с использованием пользовательских или системных учетных данных, в том числе с использованием

	программную среду инструментальные средства, необходимые ему для дальнейших действий	методов социальной инженерии
		Т3.2. Активация и выполнение вредоносного кода, внедренного в виде закладок в легитимное программное и программное- аппаратное обеспечение систем и сетей
		Т3.3. Автоматическая загрузка вредоносного кода с удаленного сайта или ресурса с последующим запуском на выполнение
Т5	Управление вредоносным программным обеспечением и (или) компонентами, к которым ранее был получен доступ Тактическая задача: внедрив вредоносное программное обеспечение или обеспечив постоянное присутствие на узле сети, нарушитель стремится автоматизировать управление внедренными инструментальными средствами, организовав взаимодействия скомпрометированным узлом и сервером управления, который может быть размещен в сети Интернет или	Т5.2. Использование штатных средств удаленного доступа и управления операционной системы
		Т5.5. Управление через съёмные носители, в частности, передача команд управления между скомпрометированными изолированной системой и подключенной к Интернет системой через носители информации, используемые на обеих системах
		Т5.9. Управление через подключенные устройства, реализующие дополнительный канал связи с внешними системами или между

	в инфраструктуре организации	скомпрометированными системами в сети
T6	Повышение привилегий по доступу к компонентам систем и сетей Тактическая задача: получив первоначальный доступ к узлу с привилегиями, недостаточными для совершения нужных ему действий, нарушитель стремится повысить полученные привилегии и получить контроль над узлом	T6.2. Подбор пароля или другой информации для аутентификации от имени привилегированной учетной записи
		T6.7. Использование уязвимостей конфигурации системы, служб и приложений, в том числе предварительно сконфигурированных профилей привилегированных пользователей, автоматически запускаемых от имени привилегированных пользователей скриптов, приложений и экземпляров окружения, позволяющих вредоносному ПО выполняться с повышенными привилегиями.
T10	Несанкционированный доступ и (или) воздействие на информационные ресурсы или компоненты систем и сетей, приводящие к негативным последствиям Тактическая задача: достижение нарушителем конечной цели, приводящее к реализации моделируемой угрозы и причинению недопустимых негативных последствий	T10.1. Несанкционированный доступ к информации в памяти системы, файловой системе, базах данных, репозиториях, в программных модулях и прошивках
		T10.3. Несанкционированное воздействие на программные модули прикладного программного обеспечения
		T10.4. Несанкционированное воздействие на программный код, конфигурацию и параметры доступа



		прикладного программного обеспечения
		Т10.8. Уничтожение информации, включая информацию, хранимую в виде файлов, информацию в базах данных и репозиториях, информацию на неразмеченных областях дисков и сменных носителей

Далее, в таблице 3 будут приведены СЗИ для нейтрализации угроз. СЗИ были взяты из [Государственного реестра сертифицированных средств защиты информации](#).

Таблица 3 – СЗИ

УБИ	СЗИ	
	№ Сертификата	Наименование средства
УБИ 6: Угроза внедрения кода или данных	3840	программное изделие «Kaspersky Security 11 для Windows Server»
УБИ 15: Угроза доступа к защищаемым файлам с использованием обходного пути	3727	программный комплекс защиты информации «ViPNet 4»
УБИ 102: Угроза опосредованного управления группой программ через совместно используемые данные	4503	программный комплекс Межсетевой экран с системой обнаружения вторжений Idecu UTM
УБИ 158: Угроза форматирования носителей информации	4596	программное обеспечение «Платформа мониторинга событий безопасности Alertix»

## **Заключение**

В процессе выполнения курсовой работы была разработана автоматизированная информационная система, основанная на учебно-исследовательской базе данных, охватывающей сферу "Казино". Пояснительная записка была структурирована в соответствии с требованиями ОС ТУСУР 01-2021. Работа над курсовой работой проводилась в соответствии с этапами, определенными в задании на лабораторную работу.

В ходе выполнения проекта были приобретены и усовершенствованы следующие навыки:

- Проектирование и создание баз данных, а также работа с ними.
- Разработка политики безопасности пользователей приложения.
- Написание SQL-запросов для взаимодействия с базой данных.
- Навыки программирования на языке C#.
- Создание приложения в среде разработки Visual Studio.

## Список использованных источников

1. ОС ТУСУР 01-2021 [Электронный ресурс]: сайт tusur.ru. URL: <https://regulations.tusur.ru/documents/70>;
2. Курс дисциплины «Безопасность систем баз данных» в системе дистанционного обучения [Электронный ресурс]: сайт sdo.tusur.ru. URL: <https://sdo.tusur.ru/course/view.php?id=2121>.
3. Вишневский, Алексей Microsoft SQL Server. Эффективная работа / Алексей Вишневский. - М.: Питер, 2019. - 847 с.
4. Курс дисциплины «Основы программирования» в системе дистанционного обучения [Электронный ресурс]: сайт sdo.tusur.ru. URL: <https://sdo.tusur.ru/course/view.php?id=13312>.
5. О персональных данных: федеральный закон от 27 июля 2006 г. N 152-ФЗ // Собрание законодательства Российской Федерации. — 2006. — № 31 (часть I). — Ст. 3451.
6. Постановление Правительства РФ от 01.11.2012 N 1119 "Об утверждении требований к защите персональных данных при их обработке в информационных системах персональных данных".
7. Приказ ФСТЭК России от 18.02.2013 N 21 (ред. от 14.05.2020) "Об утверждении Состав и содержания организационных и технических мер по обеспечению безопасности персональных данных при их обработке в информационных системах персональных данных" (Приложение Д).

## Приложение А

(обязательное)

### Листинг для главной формы

```
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;
using static System.Windows.Forms.VisualStyles.VisualStyleElement.Window;

namespace Course_Work
{
    public partial class MainForm : Form
    {
        private readonly CheckUser _user;
        public MainForm(CheckUser user)
        {
            InitializeComponent();

            StartPosition = FormStartPosition.CenterScreen;

            _user = user;

            thirdButton.Visible = _user.IsAdmin;
        }

        private void Close_button_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void Close_button_MouseEnter(object sender, EventArgs e)
        {
            Close_button.ForeColor = Color.Red;
        }

        private void Close_button_MouseLeave(object sender, EventArgs e)
        {
            Close_button.ForeColor = Color.White;
        }

        Point lastPoint;

        private void MainForm_MouseMove(object sender, MouseEventArgs e)
        {
            if (e.Button == MouseButtons.Left)
            {
                this.Left += e.X - lastPoint.X;
                this.Top += e.Y - lastPoint.Y;
            }
        }

        private void MainForm_MouseDown(object sender, MouseEventArgs e)
        {
            lastPoint = new Point(e.X, e.Y);
        }
    }
}
```

```

private void firstButton_MouseEnter(object sender, EventArgs e)
{
    firstButton.ForeColor = Color.DarkGray;
}

private void secondButton_MouseEnter(object sender, EventArgs e)
{
    secondButton.ForeColor = Color.DarkGray;
}

private void thirdButton_MouseEnter(object sender, EventArgs e)
{
    thirdButton.ForeColor = Color.DarkGray;
}

private void fourthButton_MouseEnter(object sender, EventArgs e)
{
    fourthButton.ForeColor = Color.DarkGray;
}

private void fifthButton_MouseEnter(object sender, EventArgs e)
{
    fifthButton.ForeColor = Color.DarkGray;
}

private void fifthButton_MouseLeave(object sender, EventArgs e)
{
    fifthButton.ForeColor = Color.White;
}

private void fourthButton_MouseLeave(object sender, EventArgs e)
{
    fourthButton.ForeColor = Color.White;
}

private void thirdButton_MouseLeave(object sender, EventArgs e)
{
    thirdButton.ForeColor = Color.White;
}

private void secondButton_MouseLeave(object sender, EventArgs e)
{
    secondButton.ForeColor = Color.White;
}

private void firstButton_MouseLeave(object sender, EventArgs e)
{
    firstButton.ForeColor = Color.White;
}

private void firstButton_Click(object sender, EventArgs e)
{
    Croupier croupier_table = new Croupier(_user);

    this.Hide();
    croupier_table.ShowDialog();
    this.Show();
}

private void secondButton_Click(object sender, EventArgs e)
{
    Game game = new Game(_user);

    this.Hide();

```

```

        game.ShowDialog();
        this.Show();
    }

    private void thirdButton_Click(object sender, EventArgs e)
    {
        Player player = new Player();

        this.Hide();
        player.ShowDialog();
        this.Show();
    }

    private void fourthButton_Click(object sender, EventArgs e)
    {
        PlayerInGame playerInGame = new PlayerInGame(_user);

        this.Hide();
        playerInGame.ShowDialog();
        this.Show();
    }

    private void fifthButton_Click(object sender, EventArgs e)
    {
        TransferDesk transferDesk = new TransferDesk(_user);

        this.Hide();
        transferDesk.ShowDialog();
        this.Show();
    }
}
}

```

## Приложение Б

(обязательное)

### Листинг формы с таблицей

```
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;
using System.Data.SqlClient;

namespace Course_Work
{
    enum Rowstate
    {
        Existed,
        New,
        Modified,
        Deleted,
        ModifiedNew
    }

    public partial class Game : Form
    {
        DataBase _dataBase = new DataBase();
        private readonly CheckUser _user;
        Point lastPoint;

        int selectedRow;
        public Game(CheckUser user)
        {
            InitializeComponent();
            StartPosition = FormStartPosition.CenterScreen;
            _user = user;

            delButton.Visible = user.IsAdmin;
        }

        private void Close_button_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void Close_button_MouseEnter(object sender, EventArgs e)
        {
            Close_button.ForeColor = Color.Red;
        }

        private void Close_button_MouseLeave(object sender, EventArgs e)
        {
            Close_button.ForeColor = Color.White;
        }

        private void CreateColumns()
        {
            dataGridView1.Columns.Add("game_id", "id игры");
            dataGridView1.Columns.Add("croupier_id", "id крупье");
            dataGridView1.Columns.Add("table_id", "id стола");
            dataGridView1.Columns.Add("date_time", "дата игры");
            dataGridView1.Columns.Add("best_combination", "лучшая комбинация");
        }
    }
}
```

```

        dataGridView1.Columns.Add("IsNew", "ColumnName");

        dataGridView1.Columns["IsNew"].Visible = false;
    }

    private void ReadSingleRow(DataGridView dgw, IDataRecord record)
    {
        dgw.Rows.Add(record.GetInt32(0),
            record.GetInt32(1),
            record.GetInt32(2),
            record.GetDateTime(3),
            record.GetString(4),
            Rowstate.ModifiedNew);
    }

    private void RefreshDataGrid(DataGridView dgw)
    {
        dgw.Rows.Clear();

        string queryString = $"select * from Game";

        SqlCommand cmd = new SqlCommand(queryString, _dataBase.getConnection());

        _dataBase.openConnection();

        SqlDataReader reader = cmd.ExecuteReader();

        while (reader.Read())
        {
            ReadSingleRow(dgw, reader);
        }
        reader.Close();
    }

    private void Game_Load(object sender, EventArgs e)
    {
        CreateColumns();
        RefreshDataGrid(dataGridView1);

        if (!_user.IsAdmin)
        {
            for (int i = 0; i < dataGridView1.Rows.Count; i++)
            {
                DataGridViewRow row = dataGridView1.Rows[i];
                row.Cells[3].Value = "#####";
            }
        }
    }

    private void dataGridView1_CellClick(object sender, DataGridViewCellEventArgs e)
    {
        selectedRow = e.RowIndex;

        if (e.RowIndex >= 0)
        {
            DataGridViewRow row = dataGridView1.Rows[selectedRow];

            game_id_tb.Text = row.Cells[0].Value.ToString();
            croupier_id_tb.Text = row.Cells[1].Value.ToString();
            table_id_tb.Text = row.Cells[2].Value.ToString();
            date_tb.Text = row.Cells[3].Value.ToString();
            combo_tb.Text = row.Cells[4].Value.ToString();
        }
    }

    private void button1_Click(object sender, EventArgs e)
    {

```



```

_dataBase.openConnection();

var idCroupier = croupier_id_tb.Text;
var idTable = table_id_tb.Text;
var date = date_tb.Text;
var best_comb = combo_tb.Text;

if (date != "")
{
    try
    {
        var addQuery = $"insert into Game (croupier_id, table_id, date_time, best_combination) values
('{idCroupier}', '{idTable}', '{date}', '{best_comb}')";

        var command = new SqlCommand(addQuery, _dataBase.getConnection());

        command.ExecuteNonQuery();

        MessageBox.Show("Строка добавлена в таблицу!", "Успех!", MessageBoxButtons.OK,
        MessageBoxIcon.Information);
    }
    catch (Exception)
    {
        MessageBox.Show("Проверьте правильность заполнения полей!", "Ошибка!", MessageBoxButtons.OK,
        MessageBoxIcon.Information);
        return;
    }
}
else { MessageBox.Show("Ошибка заполнения таблицы!", "Ошибка!", MessageBoxButtons.OK,
        MessageBoxIcon.Information); }
_dataBase.closeConnection();
}

private void Search(DataGridView dgv)
{
    dgv.Rows.Clear();

    string searching = $"Select * from Game where concat (croupier_id, table_id, date_time, best_combination)
like '%" + search_tb.Text + "%'";

    SqlCommand com = new SqlCommand(searching, _dataBase.getConnection());

    _dataBase.openConnection();

    SqlDataReader reader = com.ExecuteReader();

    while (reader.Read())
    {
        ReadSingleRow(dgv, reader);
    }
    reader.Close();
}

private void deleteRow()
{
    try
    {
        int index = dataGridView1.CurrentCell.RowIndex;

        dataGridView1.Rows[index].Visible = false;

        if (dataGridView1.Rows[index].Cells[0].Value.ToString() == string.Empty)
        {
            dataGridView1.Rows[index].Cells[5].Value = Rowstate.Deleted;
            return;
        }
        dataGridView1.Rows[index].Cells[5].Value = Rowstate.Deleted;
    }
}

```

```

    }
    catch(Exception) { MessageBox.Show("Ничего не выбрано!", "Ошибка!", MessageBoxButtons.OK,
    MessageBoxIcon.Information); }
    }
    new private void Update()
    {
        _dataBase.openConnection();

        for (int index = 0; index < dataGridView1.Rows.Count; index++)
        {
            var rowState = (Rowstate)dataGridView1.Rows[index].Cells[5].Value;

            if (rowState == Rowstate.Existed) continue;

            if (rowState == Rowstate.Deleted)
            {
                try
                {
                    var id = Convert.ToInt32(dataGridView1.Rows[index].Cells[0].Value);
                    var deleteQuery = $"delete from Game where game_id = {id}";

                    var command = new SqlCommand(deleteQuery, _dataBase.getConnection());
                    command.ExecuteNonQuery();
                }
                catch (Exception)
                {
                    MessageBox.Show("На данный id ссылается другая таблица!", "Ошибка!", MessageBoxButtons.OK,
    MessageBoxIcon.Information);
                    RefreshDataGrid(dataGridView1);
                }
            }

            if (rowState == Rowstate.Modified)
            {
                var id_game = dataGridView1.Rows[index].Cells[0].Value.ToString();
                var id_croupier = dataGridView1.Rows[index].Cells[1].Value.ToString();
                var id_table = dataGridView1.Rows[index].Cells[2].Value.ToString();
                var date = dataGridView1.Rows[index].Cells[3].Value.ToString();
                var combo = dataGridView1.Rows[index].Cells[4].Value.ToString();

                var changeQuery = $"update Game set croupier_id = '{id_croupier}', table_id = '{id_table}', " +
                    $"date_time = '{date}', best_combination = '{combo}' where game_id = '{id_game}'";

                var command = new SqlCommand(changeQuery, _dataBase.getConnection());
                command.ExecuteNonQuery();
            }
        }

        _dataBase.closeConnection();

        if (!_user.IsAdmin)
        {
            for (int i = 0; i < dataGridView1.Rows.Count; i++)
            {
                DataGridViewRow row = dataGridView1.Rows[i];
                row.Cells[3].Value = "#####";
            }
        }
    }

    private void search_tb_TextChanged(object sender, EventArgs e)
    {
        Search(dataGridView1);
    }

    private void delButton_Click(object sender, EventArgs e)
    {

```

```

        deleteRow();
        Update();
    }

    private void Change()
    {
        try
        {
            var selectedRowIndex = dataGridView1.CurrentCell.RowIndex;

            var IDGame = game_id_tb.Text;
            var IDCroupier = croupier_id_tb.Text;
            var IDTable = table_id_tb.Text;
            var DateTime = date_tb.Text;
            var combination = combo_tb.Text;

            if (dataGridView1.Rows[selectedRowIndex].Cells[0].Value.ToString() != string.Empty)
            {
                dataGridView1.Rows[selectedRowIndex].SetValues(IDGame, IDCroupier, IDTable, DateTime,
combination);
                dataGridView1.Rows[selectedRowIndex].Cells[5].Value = Rowstate.Modified;
            }
            else { MessageBox.Show("Ошибка заполнения таблицы!", "Ошибка!", MessageBoxButtons.OK,
MessageBoxIcon.Information); }
        }
        catch (Exception) { MessageBox.Show("Ничего не выбрано!", "Ошибка!", MessageBoxButtons.OK,
MessageBoxIcon.Information); }
    }

    private void button2_Click(object sender, EventArgs e)
    {
        RefreshDataGrid(dataGridView1);

        if (!_user.IsAdmin)
        {
            for (int i = 0; i < dataGridView1.Rows.Count; i++)
            {
                DataGridViewRow row = dataGridView1.Rows[i];
                row.Cells[3].Value = "#####";
            }
        }
    }

    private void Game_MouseMove(object sender, MouseEventArgs e)
    {
        if (e.Button == MouseButtons.Left)
        {
            this.Left += e.X - lastPoint.X;
            this.Top += e.Y - lastPoint.Y;
        }
    }

    private void Game_MouseDown(object sender, MouseEventArgs e)
    {
        lastPoint = new Point(e.X, e.Y);
    }

    private void changeButton_Click_1(object sender, EventArgs e)
    {
        Change();
        Update();
    }
}

```

## Приложение В

(обязательное)

### Листинг формы с авторизацией

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

namespace Course_Work
{
    public partial class LoginForm : Form
    {
        DataBase data = new DataBase();

        public LoginForm()
        {
            InitializeComponent();

            this.passwordBox.AutoSize = false;
            this.passwordBox.Size = new Size(passwordBox.Size.Width, 44);
            StartPosition = FormStartPosition.CenterScreen;
        }

        private void closeButton_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        private void closeButton_MouseEnter(object sender, EventArgs e)
        {
            closeButton.ForeColor = Color.Red;
        }

        private void closeButton_MouseLeave(object sender, EventArgs e)
        {
            closeButton.ForeColor = Color.White;
        }

        Point lastPoint;

        private void panel2_MouseMove(object sender, MouseEventArgs e)
        {
            if (e.Button == MouseButtons.Left)
            {
                this.Left += e.X - lastPoint.X;
                this.Top += e.Y - lastPoint.Y;
            }
        }

        private void panel2_MouseDown(object sender, MouseEventArgs e)
        {
            lastPoint = new Point(e.X, e.Y);
        }
    }
}
```

```

private void button1_Click(object sender, EventArgs e)
{
    var loginUser = loginBox.Text;
    var passUser = passwordBox.Text;

    SqlDataAdapter adapter = new SqlDataAdapter();
    DataTable dt = new DataTable();

    string querystring = $"select id_user, login_user, password_user, is_admin
from register where " +
        $"login_user = '{loginUser}' and password_user = '{passUser}'";

    SqlCommand command = new SqlCommand(querystring, data.getConnection());

    adapter.SelectCommand = command;
    adapter.Fill(dt);

    if (dt.Rows.Count == 1)
    {
        var user = new CheckUser(dt.Rows[0].ItemArray[1].ToString(),
Convert.ToBoolean(dt.Rows[0].ItemArray[3]));

        MessageBox.Show("Вы успешно вошли!", "Успешно!", MessageBoxButtons.OK,
MessageBoxIcon.Information);
        MainForm frm1 = new MainForm(user);
        this.Hide();
        frm1.ShowDialog();
        this.Show();
    }
    else MessageBox.Show("Такого аккаунта не существует!", "Аккаунта не
существует!", MessageBoxButtons.OK, MessageBoxIcon.Warning);
}

private void reg_button_Click(object sender, EventArgs e)
{
    reg_button.ForeColor = Color.Gray;

    RegForm frm2 = new RegForm();
    this.Hide();
    frm2.ShowDialog();
    this.Show();
}
}
}

```

**Приложение Г**  
**(обязательное)**  
**Выдержка из ФЗ 152**

**152 ФЗ статья 19**

Правительство Российской Федерации с учетом возможного вреда субъекту персональных данных, объема и содержания обрабатываемых персональных данных, вида деятельности, при осуществлении которого обрабатываются персональные данные, актуальности угроз безопасности персональных данных устанавливает:

- 1) [уровни защищенности](#) персональных данных при их обработке в информационных системах персональных данных в зависимости от угроз безопасности этих данных;
- 2) [требования](#) к защите персональных данных при их обработке в информационных системах персональных данных, исполнение которых обеспечивает установленные уровни защищенности персональных данных;

См. [Инструкцию](#) по организации защиты персональных данных, содержащихся в информационных системах органов внутренних дел РФ, утвержденную [приказом](#) МВД РФ от 6 июля 2012 г. N 678

- 3) [требования](#) к материальным носителям биометрических персональных данных и технологиям хранения таких данных вне информационных систем персональных данных.

4. [Состав и содержание](#) необходимых для выполнения установленных Правительством Российской Федерации в соответствии с [частью 3](#) настоящей статьи требований к защите персональных данных для каждого из уровней защищенности, организационных и технических мер по обеспечению безопасности персональных данных при их обработке в информационных системах персональных данных устанавливаются федеральным органом исполнительной власти, уполномоченным в области обеспечения безопасности, и федеральным органом исполнительной власти, уполномоченным в области противодействия техническим разведкам и технической защиты информации, в пределах их полномочий.

5. Федеральные органы исполнительной власти, осуществляющие функции по выработке государственной политики и нормативно-правовому регулированию в установленной сфере деятельности, органы государственной власти субъектов Российской Федерации, Банк России, органы государственных внебюджетных фондов, иные государственные органы в пределах своих полномочий принимают [нормативные правовые акты](#), в которых определяют угрозы безопасности персональных данных, актуальные при обработке персональных данных в информационных системах персональных данных, эксплуатируемых при осуществлении соответствующих видов деятельности, с учетом содержания персональных данных, характера и способов их обработки.

6. Наряду с угрозами безопасности персональных данных, определенных в нормативных правовых актах, принятых в соответствии с [частью 5](#) настоящей статьи, ассоциации, союзы и иные объединения операторов своими решениями вправе определить дополнительные угрозы безопасности персональных данных, актуальные при обработке персональных данных в информационных системах персональных данных, эксплуатируемых при осуществлении определенных видов деятельности членами таких ассоциаций, союзов и иных объединений операторов, с учетом содержания персональных данных, характера и способов их обработки.

7. Проекты нормативных правовых актов, указанных в [части 5](#) настоящей статьи, подлежат

согласованию с федеральным органом исполнительной власти, уполномоченным в области обеспечения безопасности, и федеральным органом исполнительной власти, уполномоченным в области противодействия техническим разведкам и технической защиты информации. Проекты решений, указанных в [части 6](#) настоящей статьи, подлежат согласованию с федеральным органом исполнительной власти, уполномоченным в области обеспечения безопасности, и федеральным органом исполнительной власти, уполномоченным в области противодействия техническим разведкам и технической защиты информации, в [порядке](#), установленном Правительством Российской Федерации. Решение федерального органа исполнительной власти, уполномоченного в области обеспечения безопасности, и федерального органа исполнительной власти, уполномоченного в области противодействия техническим разведкам и технической защиты информации, об отказе в согласовании проектов решений, указанных в части 6 настоящей статьи, должно быть мотивированным.

8. Контроль и надзор за выполнением организационных и технических мер по обеспечению безопасности персональных данных, установленных в соответствии с настоящей статьей, при обработке персональных данных в государственных информационных системах персональных данных осуществляются федеральным органом исполнительной власти, уполномоченным в области обеспечения безопасности, и федеральным органом исполнительной власти, уполномоченным в области противодействия техническим разведкам и технической защиты информации, в пределах их полномочий и без права ознакомления с персональными данными, обрабатываемыми в информационных системах персональных данных.

9. Федеральный орган исполнительной власти, уполномоченный в области обеспечения безопасности, и федеральный орган исполнительной власти, уполномоченный в области противодействия техническим разведкам и технической защиты информации, решением Правительства Российской Федерации с учетом значимости и содержания обрабатываемых персональных данных могут быть наделены полномочиями по контролю за выполнением организационных и технических мер по обеспечению безопасности персональных данных, установленных в соответствии с настоящей статьей, при их обработке в информационных системах персональных данных, эксплуатируемых при осуществлении определенных видов деятельности и не являющихся государственными информационными системами персональных данных, без права ознакомления с персональными данными, обрабатываемыми в информационных системах персональных данных.

10. Использование и хранение биометрических персональных данных вне информационных систем персональных данных могут осуществляться только на таких материальных носителях информации и с применением такой технологии ее хранения, которые обеспечивают защиту этих данных от неправомерного или случайного доступа к ним, их уничтожения, изменения, блокирования, копирования, предоставления, распространения.

11. Для целей настоящей статьи под угрозами безопасности персональных данных понимается совокупность условий и факторов, создающих опасность несанкционированного, в том числе случайного, доступа к персональным данным, результатом которого могут стать уничтожение, изменение, блокирование, копирование, предоставление, распространение персональных данных, а также иные неправомерные действия при их обработке в информационной системе персональных данных. Под уровнем защищенности персональных данных понимается комплексный показатель, характеризующий требования, исполнение которых обеспечивает нейтрализацию определенных угроз безопасности персональных данных при их обработке в информационных системах персональных данных.

Статья 19 дополнена частью 12 с 1 сентября 2022 г. - [Федеральный закон](#) от 14 июля 2022 г. N 266-ФЗ

12. Оператор обязан в [порядке](#), определенном федеральным органом исполнительной власти, уполномоченным в области обеспечения безопасности, обеспечивать взаимодействие с государственной системой обнаружения, предупреждения и ликвидации последствий

компьютерных атак на информационные ресурсы Российской Федерации, включая информирование его о компьютерных инцидентах, повлекших неправомерную передачу (предоставление, распространение, доступ) персональных данных.

Статья 19 дополнена частью 13 с 1 сентября 2022 г. - [Федеральный закон](#) от 14 июля 2022 г. N 266-ФЗ

13. Указанная в [части 12](#) настоящей статьи информация (за исключением информации, составляющей государственную тайну) передается федеральным органом исполнительной власти, уполномоченным в области обеспечения безопасности, в уполномоченный орган по защите прав субъектов персональных данных.

Статья 19 дополнена частью 14 с 1 сентября 2022 г. - [Федеральный закон](#) от 14 июля 2022 г. N 266-ФЗ

14. Порядок передачи информации в соответствии с [частью 13](#) настоящей статьи устанавливается совместно федеральным органом исполнительной власти, уполномоченным в области обеспечения безопасности, и уполномоченным органом по защите прав субъектов персональных данных.

[Федеральным законом](#) от 25 июля 2011 г. N 261-ФЗ статья 20 настоящего Федерального закона изложена в новой редакции, [распространяющейся](#) на правоотношения, возникшие с 1 июля 2011 г.  
[См. текст статьи в предыдущей редакции](#)



## Приложение Д

(обязательное)

### Выдержка из Приказа ФСТЭК 21

#### Состав и содержание

**мер по обеспечению безопасности персональных данных, необходимых для обеспечения каждого из уровней защищенности персональных данных**

I. Идентификация и аутентификация субъектов доступа и объектов доступа (ИАФ)	
ИАФ.1	Идентификация и аутентификация пользователей, являющихся работниками оператора
ИАФ.3	Управление идентификаторами, в том числе создание, присвоение, уничтожение идентификаторов
ИАФ.4	Управление средствами аутентификации, в том числе хранение, выдача, инициализация, блокирование средств аутентификации и принятие мер в случае утраты и (или) компрометации средств аутентификации
ИАФ.5	Защита обратной связи при вводе аутентификационной информации
ИАФ.6	Идентификация и аутентификация пользователей, не являющихся работниками оператора (внешних пользователей)
II. Управление доступом субъектов доступа к объектам доступа (УПД)	
УПД.1	Управление (заведение, активация, блокирование и уничтожение) учетными записями пользователей, в том числе внешних пользователей
УПД.2	Реализация необходимых методов (дискреционный, мандатный, ролевой или иной метод), типов (чтение, запись, выполнение или иной тип) и правил разграничения доступа
УПД.3	Управление (фильтрация, маршрутизация, контроль соединений, однонаправленная передача и иные способы управления) информационными потоками между устройствами, сегментами информационной системы, а также между информационными системами
УПД.4	Разделение полномочий (ролей) пользователей, администраторов и лиц, обеспечивающих функционирование информационной системы
УПД.5	Назначение минимально необходимых прав и привилегий пользователям, администраторам и лицам, обеспечивающим функционирование информационной системы
УПД.6	Ограничение неуспешных попыток входа в информационную систему (доступа к информационной системе)
УПД.13	Реализация защищенного удаленного доступа субъектов доступа к объектам доступа через внешние информационно-телекоммуникационные сети
УПД.14	Регламентация и контроль использования в информационной системе технологий беспроводного доступа
УПД.15	Регламентация и контроль использования в информационной системе мобильных технических средств

УПД.16	Управление взаимодействием с информационными системами сторонних организаций (внешние информационные системы)
V. Регистрация событий безопасности (РСБ)	
РСБ.1	Определение событий безопасности, подлежащих регистрации, и сроков их хранения
РСБ.2	Определение состава и содержания информации о событиях безопасности, подлежащих регистрации
РСБ.3	Сбор, запись и хранение информации о событиях безопасности в течение установленного времени хранения
РСБ.7	Защита информации о событиях безопасности
VI. Антивирусная защита (АВЗ)	
АВЗ.1	Реализация антивирусной защиты
АВЗ.2	Обновление базы данных признаков вредоносных компьютерных программ (вирусов)
VIII. Контроль (анализ) защищенности персональных данных (АНЗ)	
АНЗ.2	Контроль установки обновлений программного обеспечения, включая обновление программного обеспечения средств защиты информации
XI. Защита среды виртуализации (ЗСВ)	
ЗСВ.1	Идентификация и аутентификация субъектов доступа и объектов доступа в виртуальной инфраструктуре, в том числе администраторов управления средствами виртуализации
ЗСВ.2	Управление доступом субъектов доступа к объектам доступа в виртуальной инфраструктуре, в том числе внутри виртуальных машин
XII. Защита технических средств (ЗТС)	
ЗТС.3	Контроль и управление физическим доступом к техническим средствам, средствам защиты информации, средствам обеспечения функционирования, а также в помещения и сооружения, в которых они установлены, исключающие несанкционированный физический доступ к средствам обработки информации, средствам защиты информации и средствам обеспечения функционирования информационной системы, в помещения и сооружения, в которых они установлены
ЗТС.4	Размещение устройств вывода (отображения) информации, исключающее ее несанкционированный просмотр
XIII. Защита информационной системы, ее средств, систем связи и передачи данных (ЗИС)	
ЗИС.3	Обеспечение защиты персональных данных от раскрытия, модификации и навязывания (ввода ложной информации) при ее передаче (подготовке к передаче) по каналам связи, имеющим выход за пределы контролируемой зоны, в том числе беспроводным каналам связи

## Приложение Е (обязательное)

### СОГЛАСИЕ НА ОБРАБОТКУ ПЕРСОНАЛЬНЫХ ДАННЫХ

Я, \_\_\_\_\_, паспорт серия  
№ \_\_\_\_\_ выдан «\_\_» 20\_\_ г. (кем выдан) зарегистрированный по адресу: \_\_\_\_\_ даю

\_\_\_\_\_ (наименование оператора)

(ОГРН, ИНН \_\_\_\_\_), зарегистрированному по адресу:  
\_\_\_\_\_, (далее – оператор) согласие на обработку своих персональных данных.

#### **Цель обработки персональных данных:**

- *обеспечение соблюдения требований законодательства Российской Федерации;*

Перечень персональных данных, на обработку которых дается согласие:

- *фамилия, имя, отчество;*

- *реквизиты документа, удостоверяющего личность;*

- *номера телефонов;*

Перечень действий с персональными данными, на совершение которых дается согласие, общее описание используемых оператором способов обработки персональных данных: *Сбор, запись, систематизация, накопление, хранение, уточнение (обновление, изменение), извлечение, использование, передача (предоставление, доступ), обезличивание, блокирование, удаление, уничтожение персональных данных.*

Обработка вышеуказанных персональных данных будет осуществляться  
путем автоматизированной обработки персональных данных.

Даю согласие на предоставление оператором моих данных:

\_\_\_\_\_ (указать полное наименование юридического лица; фамилия, имя, отчество и адрес  
физического лица; передачу которым дается согласие)

путем \_\_\_\_\_  
(предоставления, допуска, предоставления)

Срок, в течение которого действует согласие субъекта персональных данных, а также способ его отзыва, если иное не установлено федеральным законом;

Настоящее согласие на обработку персональных данных действует с момента его представления оператору до «\_\_» \_\_\_\_\_ 24 г. и может быть отозвано мной в любое время путем подачи оператору заявления в простой письменной форме.

Персональные данные субъекта подлежат хранению согласно регламенту хранения персональной информации.

\_\_\_\_\_  
«\_\_» \_\_\_\_\_ 24 г.