

## Содержание

<b>Введение</b>	<b>6</b>
<b>1 Аналитический раздел</b>	<b>7</b>
1.1 Постановка задачи . . . . .	7
1.2 Пользователи системы . . . . .	7
1.2.1 Гость . . . . .	7
1.2.2 Аналитик . . . . .	7
1.2.3 Менеджер . . . . .	8
1.2.4 Модератор . . . . .	8
1.3 Анализ моделей баз данных . . . . .	8
1.3.1 Иерархическая база данных . . . . .	8
1.3.2 Сетевая модель базы данных . . . . .	9
1.3.3 Реляционная модель базы данных . . . . .	9
1.3.4 Выбор модели данных . . . . .	10
1.4 Анализ СУБД . . . . .	10
1.4.1 MySQL . . . . .	10
1.4.2 Microsoft SQL Server . . . . .	11
1.4.3 PostgreSQL . . . . .	12
1.4.4 Oracle . . . . .	13

1.4.5	Выбор СУБД . . . . .	13
1.5	Вывод . . . . .	13
<b>2</b>	<b>Конструкторский раздел</b>	<b>14</b>
2.1	Сценарии пользователей . . . . .	14
2.1.1	Гость . . . . .	14
2.1.2	Любой авторизованный пользователь . . . . .	14
2.1.3	Аналитик . . . . .	15
2.1.4	Менеджер . . . . .	16
2.1.5	Модератор . . . . .	18
2.2	Ролевая модель . . . . .	19
2.3	Проектирование базы данных . . . . .	21
2.3.1	Формализация сущностей системы . . . . .	21
2.3.2	Функции . . . . .	23
2.4	Проектирование приложения . . . . .	23
2.5	Вывод . . . . .	24
<b>3</b>	<b>Технологический раздел</b>	<b>25</b>
3.1	Средства реализации поставленной задачи . . . . .	25
3.2	Создание базы данных . . . . .	26
3.3	Функции . . . . .	27

3.4	Разработка компонентов . . . . .	27
3.4.1	Компонент доступа к данным . . . . .	28
3.4.2	Компонент бизнес-логики . . . . .	29
3.5	Интерфейс приложения . . . . .	29
3.6	Вывод . . . . .	33
	<b>Заключение</b>	<b>34</b>
	<b>Литература</b>	<b>35</b>
	<b>Приложение А. Создание таблиц базы данных.</b>	<b>36</b>
	<b>Приложение Б. Презентация.</b>	<b>37</b>

## **Введение**

В современном мире регулирование широко используемых в профессиональном спорте трансферов (переходов) спортсменов — это тема, к которой спортивная общественность, СМИ и государство проявляют обоснованный интерес. На протяжении последнего столетия регламентными нормами неправительственных спортивных организаций сформирован особый механизм смены спортсменами своих работодателей. Ясно, что эти особенности наиболее ярко проявляются в самых популярных и массовых командных видах спорта — футболе, хоккее, баскетболе и т. д. Очевидно, что организациям необходимы удобные приложения для проведения сделок по покупке и продаже игроков.

Целью данной работы является реализация простого в использовании и многофункционального приложения для получения информации об игроках с возможностью их покупки и продажи.

Для достижения поставленной цели необходимо решить следующие задачи:

- 1) формализовать задание, выделив соответствующих акторов и их функционал;
- 2) провести анализ СУБД и выбрать наиболее подходящую;
- 3) спроектировать базу данных;
- 4) спроектировать архитектуру приложения;
- 5) разработать приложение.

## **1 Аналитический раздел**

В данном разделе будет поставлена задача, рассмотрены возможные пользователи системы (акторы), модели данных и СУБД.

### **1.1 Постановка задачи**

Необходимо разработать программу, предоставляющую интерфейс для получения информации об игроках по заданным параметрами с возможностью их покупки и продажи.

### **1.2 Пользователи системы**

#### **1.2.1 Гость**

Гость — это неавторизованный пользователь, обладающий только одной возможностью взаимодействия с системой - авторизация.

#### **1.2.2 Аналитик**

Аналитик - это авторизированный пользователь, закрепленный за определенной командой, обладающий базовыми возможностями взаимодействия с системой - просмотр информации об игроках, их статистике за прошедший сезон и командах. Также дополнительными возможностями аналитика являются просмотр желаемых игроков для своей команды, добавление игрока в список желаемых игроков команды и удаление желаемого игрока из этого списка.

### **1.2.3 Менеджер**

Менеджер - это авторизованный пользователь, закреплённый за определённой, обладающий базовыми возможностями взаимодействия с системой - просмотр информации об игроках, их статистике за прошедший сезон и командах. Также дополнительными возможностями менеджера являются просмотр желаемых игроков для своей команды, удаление желаемого игрока из списка желаемых игроков. Менеджер может запрашивать игрока у другой команды, подтверждать и отклонять доступные сделки.

### **1.2.4 Модератор**

Модератор является авторизованным пользователем с повышенным уровнем полномочий — он обладает всеми базовыми правами пользователя, а также может проводить и удалять сделки, изменять баланс команды и команду игрока.

## **1.3 Анализ моделей баз данных**

Модель базы данных - это тип модели данных, которая определяет логическую структуру базы данных и в корне определяет, каким образом данные могут храниться, организовываться и обрабатываться.

### **1.3.1 Иерархическая база данных**

Иерархическая модель базы данных подразумевает, что элементы организованы в структуры, связанные между собой иерархическими или древовидными

связями. Родительский элемент может иметь несколько дочерних элементов. Но у дочернего элемента может быть только один предок.

Такая модель подразумевает возможность существования одинаковых (преимущественно дочерних) элементов. Данные здесь хранятся в серии записей с прикреплёнными к ним полями значений. Модель собирает вместе все экземпляры определённой записи в виде «типов записей» — они эквивалентны таблицам в реляционной модели, а отдельные записи — столбцам таблицы.

### **1.3.2 Сетевая модель базы данных**

Сетевая модель базы данных подразумевает, что у родительского элемента может быть несколько потомков, а у дочернего элемента — несколько предков. Записи в такой модели связаны списками с указателями.

Сетевая модель позволяет иметь несколько предков и потомков, формирующих решётчатую структуру.

### **1.3.3 Реляционная модель базы данных**

В реляционной модели, в отличие от иерархической или сетевой, не существует физических отношений. Вся информация хранится в виде таблиц (отношений), состоящих из рядов и столбцов. А данные двух таблиц связаны общими столбцами, а не физическими ссылками или указателями. Для манипуляций с рядами данных существуют специальные операторы.

Реляционные таблицы обладают следующими свойствами:

- 1) все значения атомарны;
- 2) каждый ряд уникален;
- 3) порядок столбцов не важен;
- 4) порядок рядов не важен;
- 5) у каждого столбца есть своё уникальное имя.

#### **1.3.4 Выбор модели данных**

В качестве модели данных была выбрана реляционная модель, так как структура данных однозначно определена, структура данных не является быстроизменяющейся и данные подчиняются строгим правилам и ограничениям.

### **1.4 Анализ СУБД**

СУБД — комплекс программ, позволяющих создать базу данных (БД) и манипулировать данными (вставлять, обновлять, удалять и выбирать). Система обеспечивает безопасность, надёжность хранения и целостность данных, а также предоставляет средства для администрирования БД. Самыми популярными СУБД являются MySQL, Microsoft SQL Server, PostgreSQL и Oracle.

#### **1.4.1 MySQL**

MySQL — реляционная СУБД с открытым исходным кодом, главными плюсами которой являются ее скорость и гибкость, которая обеспечена поддержкой большого количества различных типов таблиц.



Кроме того, это надежная бесплатная система с простым интерфейсом и возможностью синхронизации с другими базами данных. В совокупности эти факторы позволяют использовать MySQL как крупным корпорациям, так и небольшим компаниям.

Преимущества:

- 1) простота в использовании;
- 2) обширный функционал;
- 3) безопасность;
- 4) масштабируемость;
- 5) скорость.

Недостатки:

- 1) недостаточная надежность;
- 2) низкая скорость разработки.

#### **1.4.2 Microsoft SQL Server**

Система позволяет синхронизироваться с другими программными продуктами компании Microsoft, а также обеспечивает надежную защиту данных и простой интерфейс, однако отличается высокой стоимостью лицензии и повышенным потреблением ресурсов.

Преимущества:

- 1) СУБД масштабируется;
- 2) простота в использовании;
- 3) возможность интеграции с другими продуктами Microsoft.

Недостатки:

- 1) высокая стоимость продукта для юридических лиц;
- 2) возможны проблемы в работе служб интеграции импорта файлов;
- 3) высокая ресурсоемкость SQL Server.

### **1.4.3 PostgreSQL**

PostgreSQL — это популярная свободная объектно-реляционная система управления базами данных. PostgreSQL базируется на языке SQL и поддерживает многочисленные возможности.

Преимущества:

- 1) бесплатное ПО с открытым исходным кодом;
- 2) большое количество дополнений;
- 3) расширения;

Недостатки:

- 1) производительность;
- 2) популярность;

#### **1.4.4 Oracle**

Oracle – это объектно-реляционная система управления базами данных.

Преимущества:

- 1) поддержка огромных баз данных;
- 2) быстрая обработка транзакций;
- 3) большой и постоянно развивающийся функционал.

Недостатки:

- 1) высокая стоимость;
- 2) значительные вычислительные ресурсы.

#### **1.4.5 Выбор СУБД**

Для данной задачи PostgreSQL[1] выигрывает по многим параметрам и поэтому было решено использовать именно эту СУБД.

### **1.5 Вывод**

В данном разделе была поставлена задача, рассмотрены возможные пользователи системы, проанализированы модели данных и СУБД.

## 2 Конструкторский раздел

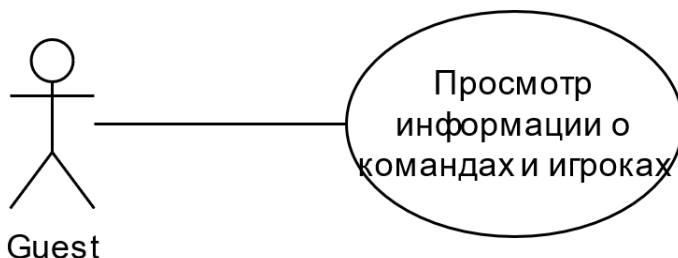
В данном разделе будут рассмотрены сценарии пользователей, спроектирована база данных, описана ролевая модель и спроектировано приложение.

### 2.1 Сценарии пользователей

Необходимо определить функционал каждого из пользователей.

#### 2.1.1 Гость

Гость - это неавторизованный пользователь, у которого есть только одна возможность - авторизоваться. На рисунке 1 представлена Use-Case-диаграмма для гостя.



**Рисунок 1** – Сценарии для гостя.

#### 2.1.2 Любой авторизованный пользователь

Функциональность, предоставляемая каждому авторизованного пользователя (аналитику, менеджеру, модератору):

- 1) просмотр всех игроков;

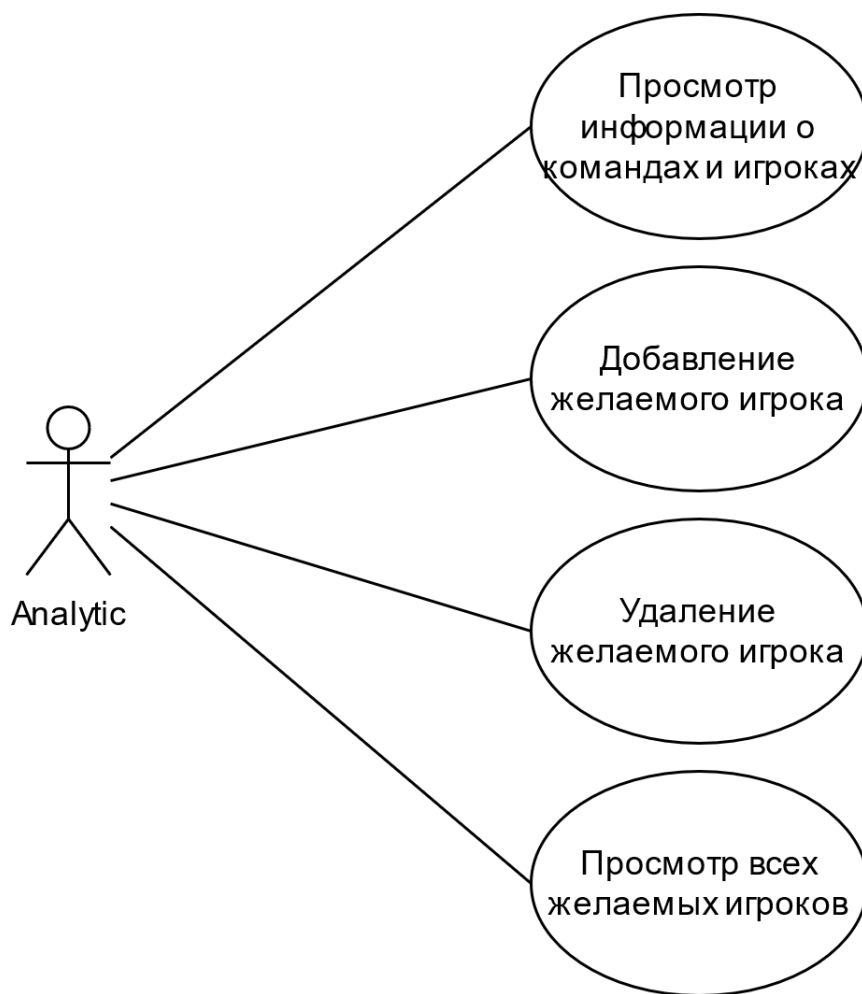
- 2) просмотр статистики игроков;
- 3) получение игроков выбранной команды;
- 4) поиск игрока по ID;
- 5) поиск игрока по имени;
- 6) просмотр всех команд;
- 7) поиск команды по ID;
- 8) поиск команды по имени;
- 9) получение статистики выбранного игрока.

### **2.1.3 Аналитик**

Дополнительная функциональность, предоставляемая пользователю аналитик:

- 1) добавить желаемого игрока;
- 2) удалить желаемого игрока;
- 3) просмотр всех желаемых игроков.

На рисунке 2 представлена Use-Case-диаграмма для аналитика.



**Рисунок 2** – Сценарии для аналитика.

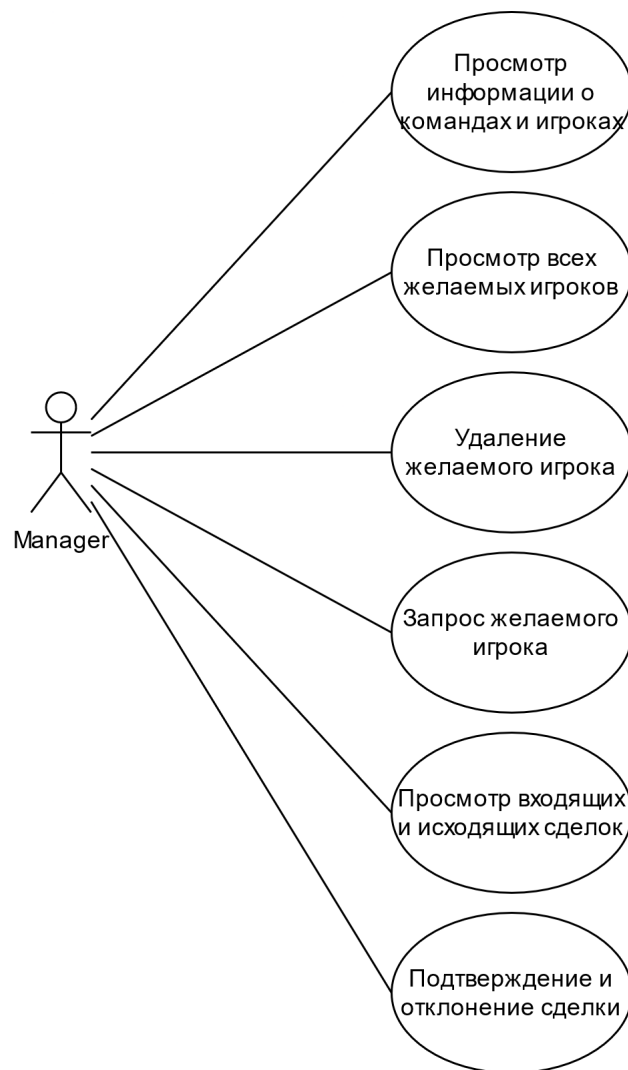
#### **2.1.4 Менеджер**

Дополнительная функциональность, предоставляемая пользователю аналитик:

- 1) просмотр всех желаемых игроков;
- 2) удалить желаемого игрока;
- 3) запросить желаемого игрока с указанием предлагаемой цены;
- 4) подтвердить сделку;

- 5) отклонить сделку;
- 6) получить входящие предложения;
- 7) получить исходящие предложения.

На рисунке 3 представлена Use-Case-диаграмма для менеджера.



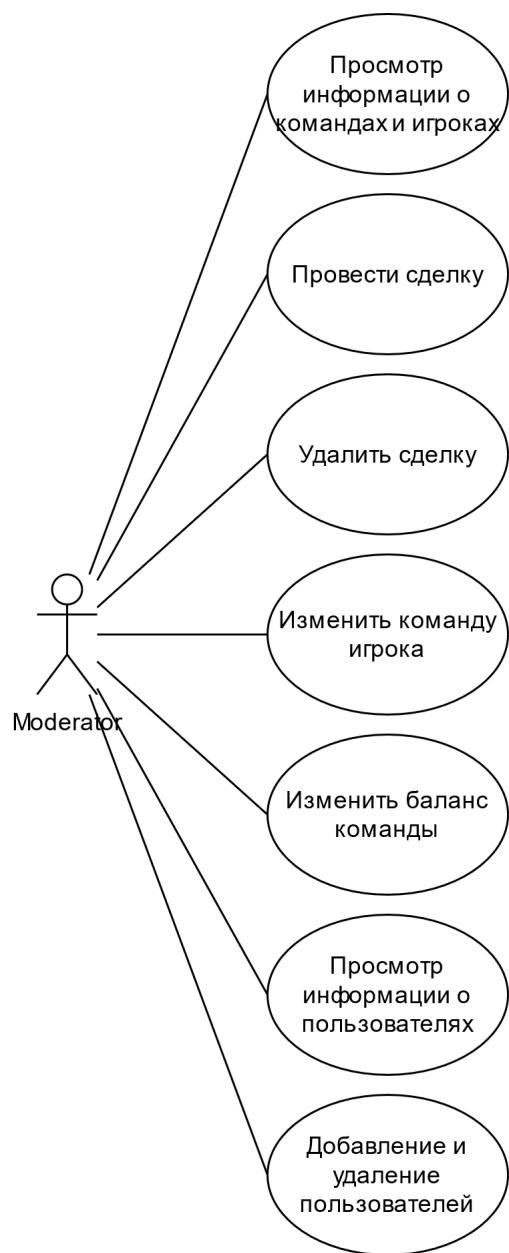
**Рисунок 3 – Сценарии для менеджера.**

### **2.1.5 Модератор**

- 1) провести сделку;
- 2) удалить сделку;
- 3) изменить команду игрока;
- 4) изменить баланс команды;
- 5) просмотр всех пользователей;
- 6) добавление нового пользователя;
- 7) удаление пользователя.

На рисунке 4 представлена Use-Case-диаграмма для модератора.





**Рисунок 4 – Сценарии для модератора.**

## **2.2 Ролевая модель**

На уровне базы данных выделена следующая ролевая модель:

- 1) guest - гость;
- 2) analytic - аналитик;

- 3) manager - менеджер;
- 4) moderator - модератор.

Использование ролевой модели на уровне базы данных гарантирует безопасность доступа к объектам базы данных.

### **Гость**

Пользователь с ролью guest имеет следующие права: SELECT над таблицей UserInfo.

### **Аналитик**

Пользователь с ролью analytic имеет следующие права:

- 1) SELECT над таблицей Player;
- 2) SELECT над таблицей Management;
- 3) SELECT над таблицей Team;
- 4) SELECT над таблицей Statistics;
- 5) SELECT, UPDATE, DELETE, INSERT над таблицей DesiredPlayers.

### **Менеджер**

Пользователь с ролью manager имеет следующие права:

- 1) SELECT над таблицей Player;
- 2) SELECT над таблицей Management;
- 3) SELECT над таблицей Team;

4) SELECT над таблицей Statistics;

5) SELECT, DELETE над таблицей DesiredPlayers.

## **Модератор**

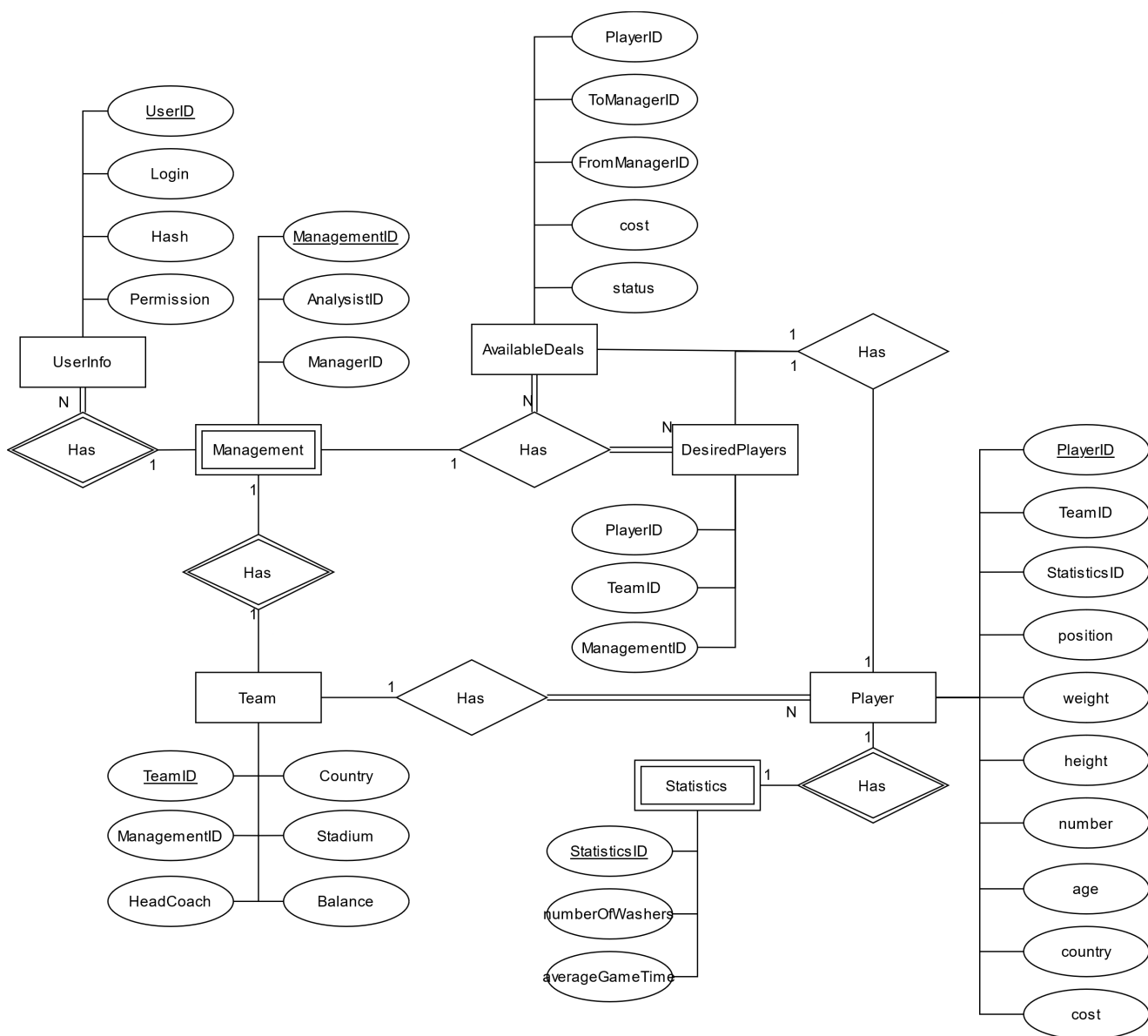
Пользователь с ролью moderator обладает всеми правами.

## **2.3 Проектирование базы данных**

### **2.3.1 Формализация сущностей системы**

Необходимо выделить сущности предметной области и построить ER-диаграмму.

На рисунке 5 представлена ER-диаграмма системы.



**Рисунок 5 – ER-диаграмма системы.**

Выделенные сущности:

- 1) User - таблица, в которой хранится информация о пользователях;
- 2) Management - таблица, в которой хранятся ID аналитиков и менеджеров, которые закреплены за определенной командой;
- 3) Team - таблица, в которой хранится информация о командах;

- 4) Player - таблица, в которой хранится информация об игроках;
- 5) AvailableDeals - таблица, в которой хранится информация о доступных сделках определенного менеджмента;
- 6) DesiredPlayers - таблица, в которой хранится информация о желаемых игроках определенного менеджмента;
- 7) Statistics - таблица, в которой хранится информация о статистике определенного игрока.

### **2.3.2 Функции**

Для того, чтобы пользователь мог одновременно просматривать информацию об игроке, его команде и его статистику, необходимо добавить функцию, которая возвращает эту информацию.

## **2.4 Проектирование приложения**

Приложение для работы с базой данных представляет собой Desktop-приложение. Оно спроектировано по паттерну MVC. Выделены два компонента - доступа к данным и бизнес-логики. Компонент доступа к данным спроектирован по паттерну "Репозиторий". Для контроля ролей при авторизации пользователя создан класс Connection, в котором обрабатывается конфигурационный файл и возвращается строка подключения.

## **2.5 Вывод**

В данном разделе были рассмотрены сценарии пользователей, спроектирована база данных, описана ролевая модель и спроектировано приложение.

### **3 Технологический раздел**

В данном разделе будут выбраны средства реализации поставленной задачи, создана база данных и ролевая модель, разработаны компоненты и описан порядок работы.

#### **3.1 Средства реализации поставленной задачи**

В качестве языка программирования был выбран язык C#[2], так как:

- 1) этот язык имеет удобные пакеты для работы с PostgreSQL;
- 2) данный язык программирования объектно-ориентирован, что позволит использовать наследование, интерфейсы, абстракции и т.д.

В качестве среды разработки была выбрана "Microsoft Visual Studio 2019"[3], поскольку:

- 1) она имеет множество удобств для написания и отладки кода;
- 2) является бесплатной для студентов;
- 3) обеспечивает работу с Windows Forms[4].

Для работы с базой данных был выбран Entity Framework[5], так как EF Core поддерживает запросы LINQ, отслеживание изменений, обновления и миграции схемы и работает с многими базами данных, включая PostgreSQL.

### 3.2 Создание базы данных

Необходимо построить диаграмму БД по выделенным сущностям. На рисунке 6 представлена диаграмма БД. В приложении А приведен листинг создания таблиц БД.

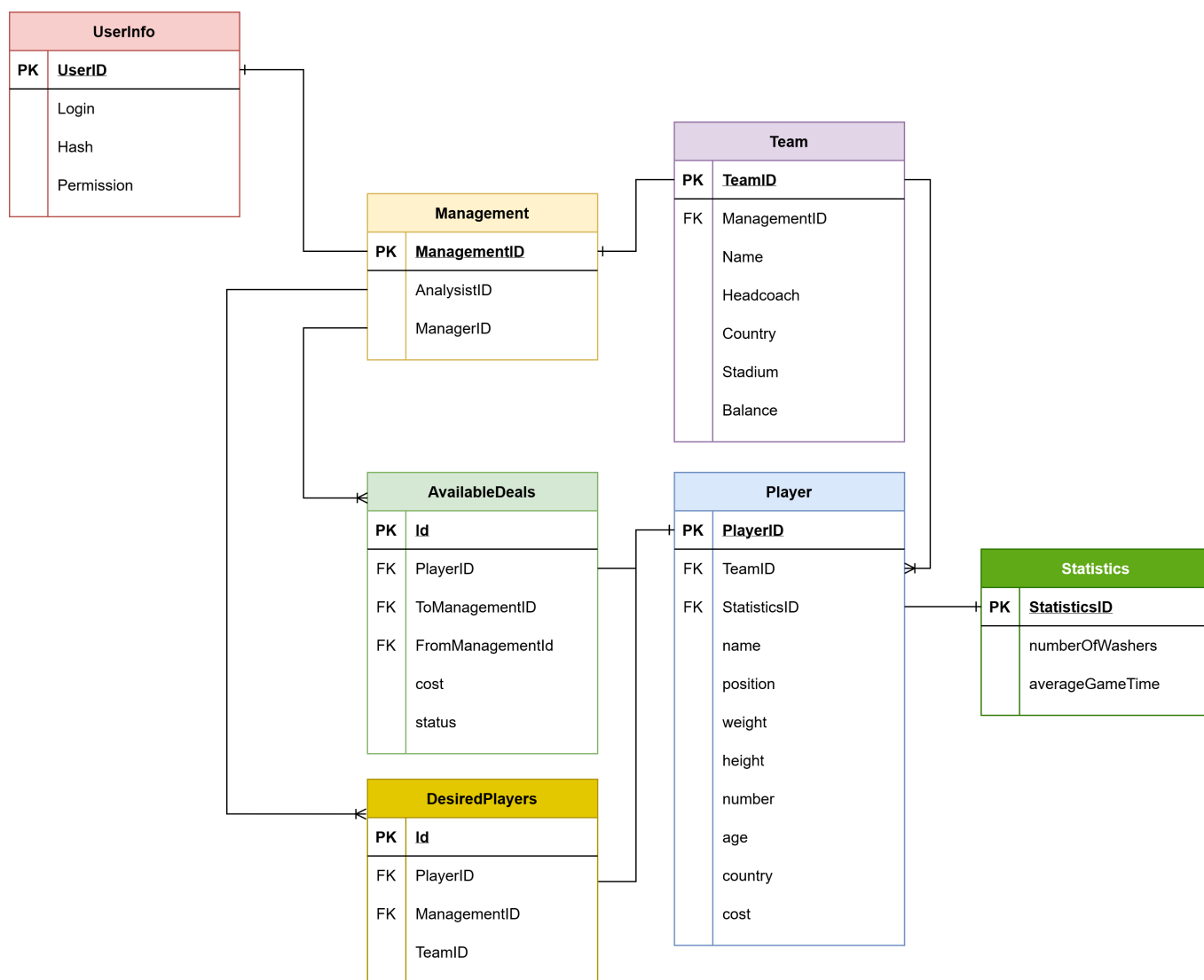


Рисунок 6 – Диаграмма БД.



### 3.3 Функции

В листинге 1 представлена реализация функции GetPlayers.

**Листинг 1** – Реализация функции GetPlayers.

```
0 drop function if exists GetPlayers;
1 create function GetPlayers()
2 returns table
3 (
4     PlayerID int,
5     Player varchar(40),
6     Team varchar(40),
7     Washers int,
8     gametime int
9 )
10 language sql
11 as $$
12     select PlayerTeam.playerid as PlayerID, PlayerTeam.player as Player,
13           PlayerTeam.Team as Team, statistics.numberOfWorkers as Washers,
14           statistics.averageGameTime as gametime
15 from (
16     select PlayerID as playerid, player.Name as player,
17           player.Statistics as PlayerStat, team.Name as Team
18     from player join team on player.TeamID = team.TeamID
19 ) as PlayerTeam
20 join statistics on PlayerTeam.PlayerStat = statistics.StatisticsID;
21 $$;
```

### 3.4 Разработка компонентов

Так как приложение спроектировано по паттерну MVC, необходимо реализовать компоненты: доступа к данным и бизнес-логики.

### 3.4.1 Компонент доступа к данным

На рисунке 7 представлена UML-диаграмма компонента доступа к данным.

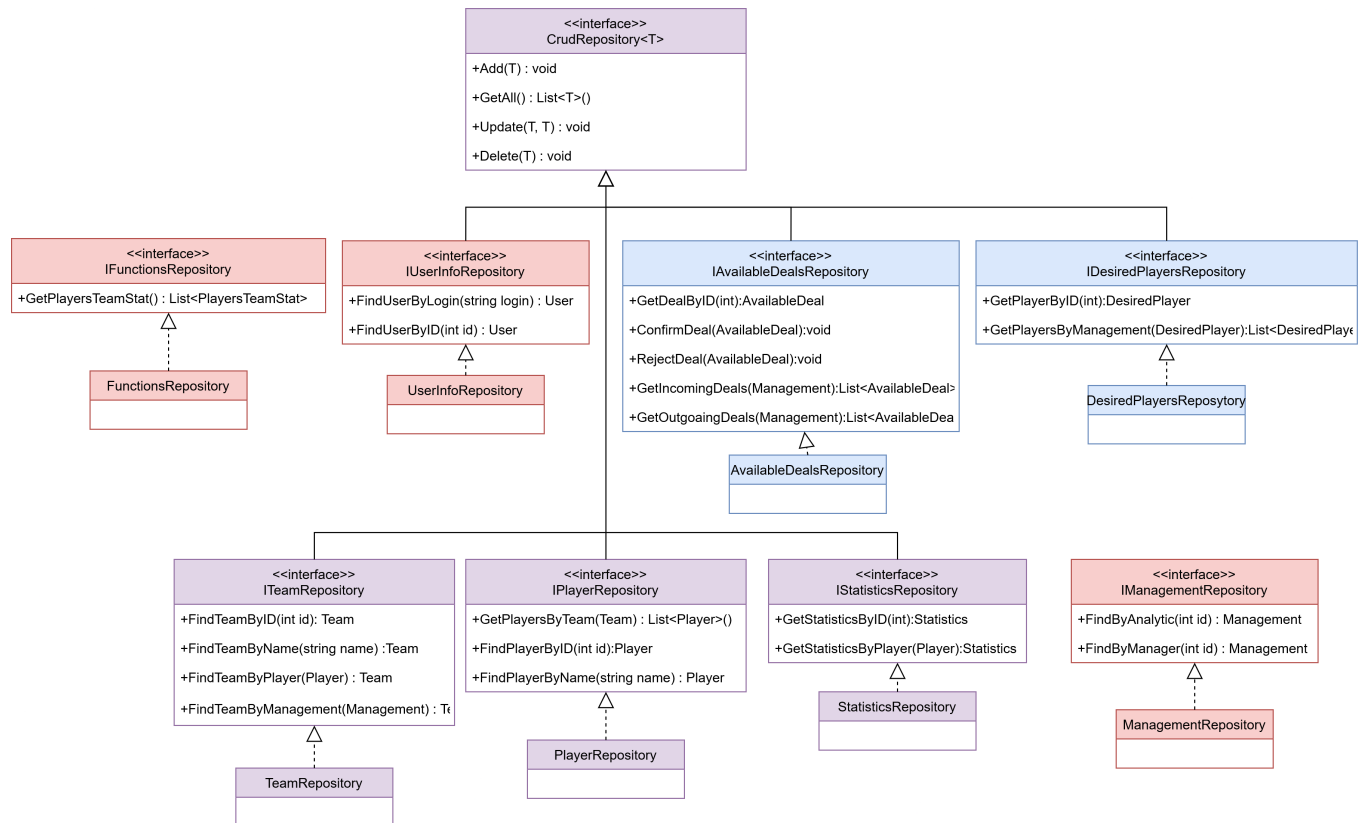


Рисунок 7 – Компонент доступа к данным.

### 3.4.2 Компонент бизнес-логики

На рисунке 8 представлена UML-диаграмма компонента бизнес-логики.

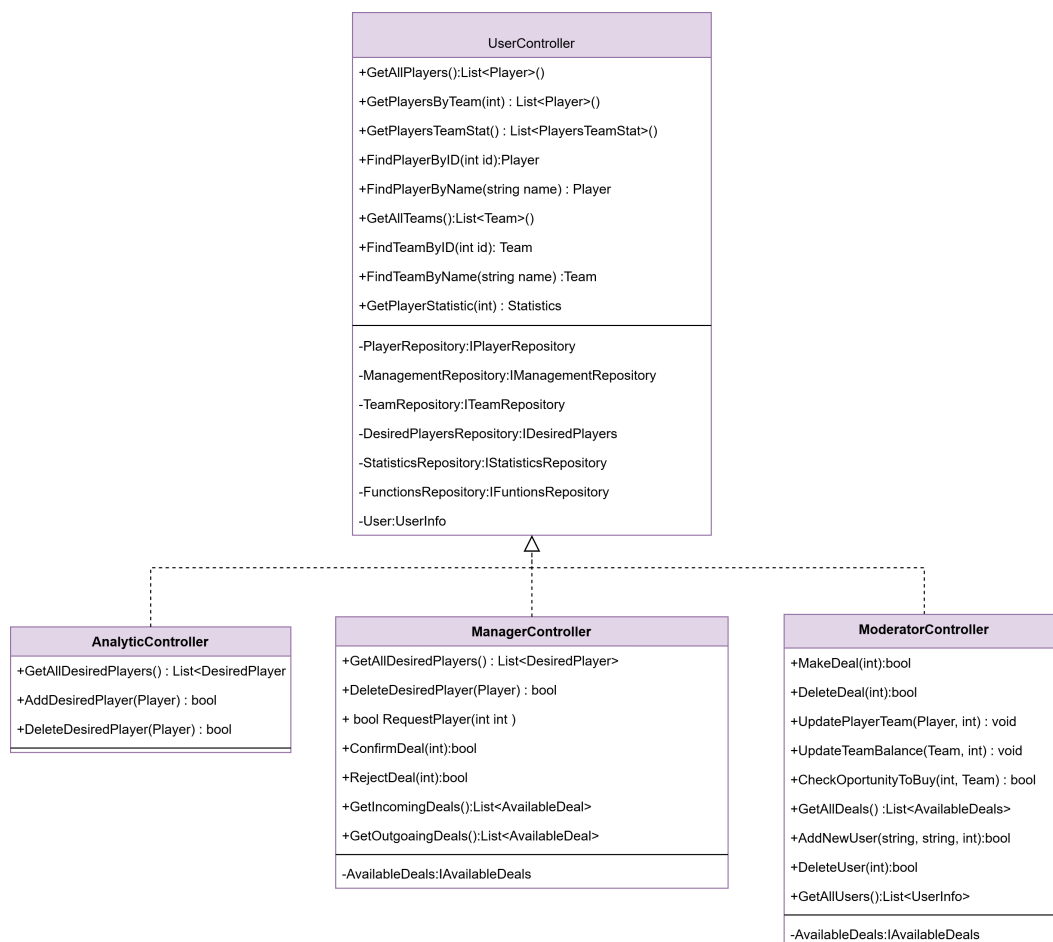


Рисунок 8 – Компонент бизнес-логики.

### 3.5 Интерфейс приложения

На рисунках ниже показан интерфейс авторизации и интерфейсы пользователей (аналитик, менеджер, модератор).

The screenshot shows a window titled "Form1" with a standard Windows title bar (minimize, maximize, close buttons). Inside the window, there are two text input fields. The first is labeled "Login" and the second is labeled "Password". Below these fields is a button labeled "Log in".

**Рисунок 9 – Окно авторизации.**

The screenshot shows a complex application window titled "Form1". The interface is divided into several sections:

- User Section:** Contains buttons for "Просмотреть всех игроков" (View all players), "Получить игрока по ID" (Get player by ID), "Получить игрока по Name" (Get player by Name), and "Получить игроков, их команды и статистику" (Get players, their teams and statistics). Below these are fields for "Team ID", "Team Name", and "Statistics ID" with corresponding "Получить" (Get) buttons.
- Analytic Section:** Contains buttons for "Получить желаемых игроков" (Get desired players), "Добавить желаемого игрока" (Add desired player), and "Удалить желаемого игрока" (Delete desired player). It also has fields for "Player ID", "DesiredPlayerID", "Cost", and "Deal ID" with buttons for "Запросить игрока" (Request player), "Подтвердить сделку" (Confirm deal), and "Отклонить сделку" (Reject deal).
- Manager Section:** Contains buttons for "Получить желаемых игроков" (Get desired players) and "Удалить желаемого игрока" (Delete desired player). It has fields for "Player ID" and "DesiredPlayerID".
- Moderator Section:** Contains buttons for "Провести сделку" (Execute deal), "Удалить сделку" (Delete deal), "Получить все сделки" (Get all deals), and "Просмотреть всех пользователей" (View all users). It has fields for "DealID", "Username", "Password", "Permission", and "User ID" with buttons for "Добавить нового пользователя" (Add new user) and "Удалить пользователя" (Delete user).
- Information Section:** Displays user information: "Информация о пользователе: Login: admin, Password: admin, Permission: Модератор, ID: 101".

**Рисунок 10 – Основное окно.**

User

Player

Просмотреть всех игроков

Player ID

Получить игрока по ID

Player Name

Получить игроков по Name

Получить игроков, их команды и статистики

Find players by team

Team ID

Получить игроков по ID

Team

Просмотреть все команды

Team ID

Получить команду по ID

Team Name

Получить команду по Name

Statistics

Statistics ID

Получить статистику

Analytic

Получить желаемых игроков

Player ID

Добавить желаемого игрока

DesiredPlayerID

Удалить желаемого игрока

Manager

Получить желаемых игроков

Player ID

Cost

Запросить игрока

Deal ID

Подтвердить сделку

Отклонить сделку

Получить входящие предложения

Получить исходящие предложения

DesiredPlayerID

Удалить желаемого игрока

Moderator

DealID

Провести сделку

Удалить сделку

Получить все сделки

Просмотреть всех пользователей

Username

Password

Permission

Добавить нового пользователя

User ID

Удалить пользователя

Информация о пользователе:

Login:

1

Password:

1

Permission:

Аналитик

ID

102

Рисунок 11 – Операции, доступные аналитику.

<div>User</div> <div>Player</div> <div>Просмотреть всех игроков</div> <div>Player ID <input type="text"/></div> <div>Получить игрока по ID</div> <div>Player Name <input type="text"/></div> <div>Получить игроков по Name</div> <div>Получить игроков, их команды и статистики</div> <div>Find players by team</div> <div>Team ID <input type="text"/></div> <div>Получить игроков по ID</div> <div>Team</div> <div>Просмотреть все команды</div> <div>Team ID <input type="text"/></div> <div>Получить команду по ID</div> <div>Team Name <input type="text"/></div> <div>Получить команду по Name</div> <div>Statistics</div> <div>Statistics ID <input type="text"/></div> <div>Получить статистику</div>	<div>Analytic</div> <div>Получить желаемых игроков</div> <div>Player ID <input type="text"/></div> <div>Добавить желаемого игрока</div> <div>DesiredPlayerID <input type="text"/></div> <div>Удалить желаемого игрока</div> <div>Manager</div> <div>Получить желаемых игроков</div> <div>Player ID <input type="text"/></div> <div>Cost <input type="text"/></div> <div>Запросить игрока</div> <div>Deal ID <input type="text"/></div> <div>Подтвердить сделку</div> <div>Отклонить сделку</div> <div>Получить входящие предложения</div> <div>Получить исходящие предложения</div> <div>DesiredPlayerID <input type="text"/></div> <div>Удалить желаемого игрока</div>	<div>Moderator</div> <div>DealID <input type="text"/></div> <div>Провести сделку</div> <div>Удалить сделку</div> <div>Получить все сделки</div> <div>Просмотреть всех пользователей</div> <div>Username <input type="text"/></div> <div>Password <input type="text"/></div> <div>Permission <input type="text"/></div> <div>Добавить нового пользователя</div> <div>User ID <input type="text"/></div> <div>Удалить пользователя</div> <div>Информация о пользователе:</div> <div>Login: 2</div> <div>Password: 2</div> <div>Permission: Менеджер</div> <div>ID 103</div>
---	---	---

Рисунок 12 – Операции, доступные менеджеру.

User	Analytic	Moderator
<b>Player</b> <input type="button" value="Просмотреть всех игроков"/> Player ID <input type="text"/> <input type="button" value="Получить игрока по ID"/> Player Name <input type="text"/> <input type="button" value="Получить игроков по Name"/> <input type="button" value="Получить игроков, их команды и статистики"/>	<input type="button" value="Получить желаемых игроков"/> Player ID <input type="text"/> <input type="button" value="Добавить желаемого игрока"/> DesiredPlayerID <input type="text"/> <input type="button" value="Удалить желаемого игрока"/>	DealID <input type="text"/> <input type="button" value="Провести сделку"/> <input type="button" value="Удалить сделку"/> <input type="button" value="Получить все сделки"/> <input type="button" value="Просмотреть всех пользователей"/>
<b>Find players by team</b> Team ID <input type="text"/> <input type="button" value="Получить игроков по ID"/>	<b>Manager</b> <input type="button" value="Получить желаемых игроков"/> Player ID <input type="text"/> Cost <input type="text"/> <input type="button" value="Запросить игрока"/> Deal ID <input type="text"/> <input type="button" value="Подтвердить сделку"/> <input type="button" value="Отклонить сделку"/> <input type="button" value="Получить входящие предложения"/> <input type="button" value="Получить исходящие предложения"/> DesiredPlayerID <input type="text"/> <input type="button" value="Удалить желаемого игрока"/>	Username <input type="text"/> Password <input type="text"/> Permission <input type="text"/> <input type="button" value="Добавить нового пользователя"/> User ID <input type="text"/> <input type="button" value="Удалить пользователя"/>
<b>Team</b> <input type="button" value="Просмотреть все команды"/> Team ID <input type="text"/> <input type="button" value="Получить команду по ID"/> Team Name <input type="text"/> <input type="button" value="Получить команду по Name"/>		<b>Информация о пользователе:</b> Login: admin Password: admin Permission: Модератор ID 101
<b>Statistics</b> Statistics ID <input type="text"/> <input type="button" value="Получить статистику"/>		

**Рисунок 13** – Операции, доступные модератору.

### 3.6 Вывод

В данном разделе были выбраны средства реализации поставленной задачи, создана база данных и описана ролевая модель на уровне БД, разработаны компоненты и описан порядок работы.

## Заключение

Цель курсовой работы достигнута.

В ходе выполнения курсовой работы было формализовано задание, выделены соответствующие акторы и их функционал, проведен анализ СУБД и выбрана наиболее подходящая для данной задачи, спроектирована база данных, архитектура приложения.

В результате, с использованием языка программирования C# и СУБД PostgreSQL было создано многофункциональное приложение для получения информации об игроках с возможностью их покупки и продажи. Получен опыт разработки базы данных и приложения по паттерну MVC.

В дальнейшей перспективе приложение и база данных могут быть масштабированы. Может быть добавлен следующий функционал:

- 1) возможность добавления модератором игроков и команд;
- 2) возможность добавления нескольких лиг для команд и перехода игроков по лигам;
- 3) могут быть добавлены другие акторы, например игрок, менеджер игрока и т.д.



## Список литературы

- [1] PostgreSQL : Документация [Электронный ресурс] URL: <https://postgrespro.ru/docs/postgresql> (дата обращения: 20.05.2021).
- [2] Документация по C# [Электронный ресурс] URL: <https://docs.microsoft.com/ru-ru/dotnet/csharp/> (дата обращения: 20.05.2021).
- [3] Документация по семейству продуктов Visual Studio [Электронный ресурс] URL: <https://docs.microsoft.com/ru-ru/visualstudio/?view=vs-2019> (дата обращения: 20.05.2021).
- [4] Windows Forms [Электронный ресурс] URL: <https://docs.microsoft.com/ru-ru/dotnet/desktop/winforms/?view=netdesktop-5.0> (дата обращения: 20.05.2021).
- [5] Документация по Entity Framework [Электронный ресурс] URL: <https://docs.microsoft.com/ru-ru/ef/> (дата обращения: 20.05.2021).

## Приложение А.

### Создание таблиц базы данных.

```
0 create table if not exists UserInfo (  
1     id int not null primary key,  
2     login varchar(30) not null,  
3     hash varchar(30) not null,  
4     permission int not null  
5 );  
6  
7 create table if not exists Management (  
8     ManagementID int not null primary key,  
9     AnalystID int references UserInfo(id),  
10    ManagerID int references UserInfo(id)  
11 );  
12  
13 create table if not exists Team (  
14     TeamID int not null primary key,  
15     ManagementID int references Management(ManagementID),  
16     Name varchar(30) not null,  
17     HeadCoach varchar(30) not null,  
18     Country varchar(30) not null,  
19     Stadium varchar(60) not null,  
20     Balance int not null  
21 );  
22  
23 create table if not exists Statistics (  
24     StatisticsID int not null primary key,  
25     numberOfWashers int not null,  
26     averageGameTime int not null  
27 );  
28  
29 create table if not exists Player (  
30     PlayerID int not null primary key,  
31     TeamID int references Team(TeamID),
```

```

32     Statistics int references Statistics(StatisticsID),
33     name varchar(30) not null,
34     position varchar(10) not null,
35     weight int not null,
36     height int not null,
37     number int not null,
38     age int not null,
39     country varchar(50) not null,
40     cost int not null
41 );
42
43 create table if not exists AvailableDeals (
44     Id int primary key not null,
45     PlayerID int references Player(PlayerID),
46     ToManagementID int references Management(ManagementID),
47     FromManagementID int references Management(ManagementID),
48     cost int not null,
49     status int not null
50 );
51
52 create table if not exists DesiredPlayers (
53     Id int primary key not null,
54     PlayerID int references Player(PlayerID),
55     TeamID int references Team(TeamID),
56     ManagementID int references Management(ManagementID)
57 );

```