

**Министерство науки и высшего образования Российской Федерации**  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
**АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**  
**(ФГБОУ ВО «АмГУ»)**

Факультет математики и информатики  
Кафедра информационных и управляющих систем  
Направление подготовки /специальность 09.04.04 – Программная инженерия  
Направленность (профиль) образовательной программы Управление разработкой программного обеспечения

ДОПУСТИТЬ К ЗАЩИТЕ

Зав. кафедрой

\_\_\_\_\_ А.В. Бушманов

«\_\_\_\_\_» \_\_\_\_\_ 2023 г.

**МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ**

на тему: Разработка программы информационной поддержки киберспортивных команд с использованием алгоритма MapReduce

Исполнитель

студент группы 157-ом

\_\_\_\_\_

(подпись, дата)

Р.К. Сергеев

Руководитель

доцент, канд. техн. наук

\_\_\_\_\_

(подпись, дата)

Т.А. Галаган

Руководитель научного

содержания программы

магистратуры

профессор, доктор техн.

наук

\_\_\_\_\_

(подпись, дата)

И.Е. Ерёмин

Нормоконтроль

доцент, канд. техн. наук

\_\_\_\_\_

(подпись, дата)

Л.В. Никифорова

Рецензент

\_\_\_\_\_

(подпись, дата)

\_\_\_\_\_

Благовещенск 2023

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
**АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**  
(ФГБОУ ВО «АмГУ»)

Факультет математики и информатики  
Кафедра информационных и управляющих систем

УТВЕРЖДАЮ  
Зав. кафедрой  
\_\_\_\_\_ А.В. Бушманов  
« \_\_\_\_\_ » \_\_\_\_\_ 2023 г.

### ЗАДАНИЕ

К выпускной квалификационной работе студента Сергеева Романа Константиновича

1. Тема выпускной квалификационной работы: Разработка программы информационной поддержки киберспортивных команд с использованием алгоритма MapReduce

(утверждена приказом от \_\_\_\_\_ № \_\_\_\_\_)

2. Срок сдачи студентами законченной работы (проекта): 23.06.2023 г.

3. Исходные данные к выпускной квалификационной работе: предметная область, отчеты по практической подготовке, результаты выступления на научной конференции

4. Содержание выпускной квалификационной работы (перечень подлежащих разработке вопросов): анализ предметной области проводимого исследования, разработка и реализация алгоритма сбора данных с использованием MapReduce, создание модели машинного обучения

5. Дата выдачи задания: 31.01.2023 г.

6. Руководитель выпускной квалификационной работы: Т.А. Галаган, доцент, канд. техн. наук

Задание принял к исполнению (31.01.2023) \_\_\_\_\_

## РЕФЕРАТ

Магистерская диссертация содержит 115 с., 64 рисунка, 21 таблицу, 1 приложение, 69 источников.

WEB-PARSING, БОЛЬШИЕ ДАННЫЕ, АЛГОРИТМ СБОРА ДАННЫХ, АРХИТЕКТУРА ИС, МАШИННОЕ ОБУЧЕНИЕ, РАЗРАБОТКА ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА, ТЕСТИРОВАНИЕ, АНАЛИЗ ПРАКТИЧЕСКОЙ ЗНАЧИМОСТИ

Целью работы является использование алгоритма обработки больших данных MapReduce в разработке программы информационной поддержки киберспортивных команд.

В данной работе описываются алгоритмы обработки больших данных, а также алгоритмы сбора данных, необходимые для корректной работы программного продукта. Подробно описано разработанная база данных, в которую записывается информация, собранная с помощью Web-parsing. На основе собранных данных происходит обучение модели машинного обучения. Создан удобный интерфейс для работы с программными ресурсами продукта.

Задачи выпускной работы:

- анализ киберспортивной дисциплины «DOTA 2»;
- описание алгоритма обработки больших данных MapReduce;
- проектирование базы данных;
- разработка алгоритмов для сбора данных с использованием MapReduce и для создания модели машинного обучения;
- выбор архитектуры программного продукта;
- обоснование выбора программно-технического обеспечения;
- разработка серверной и клиентской части;
- тестирование и отладка разработанного программного продукта.

## СОДЕРЖАНИЕ

Введение	6
1 Общая характеристика исследуемой задачи	8
1.1 Краткие сведения о киберспорте	8
1.2 Главная киберспортивная дисциплина – DOTA 2	10
1.3 Рассмотрение существующих методик решения поставленной научной задачи	11
1.3.1 Технологии и тенденции работы с Big Data	11
1.3.2 Методы и техники анализа больших данных	12
1.3.3 Аналогии разрабатываемого проекта	14
2 Алгоритмическое и программное обеспечение решения задачи	20
2.1 Синтез алгоритма сбора данных	20
2.1.1 Теоретическая информация об алгоритме сбора данных	20
2.1.2 Автоматизированный алгоритм для сбора данных	28
2.2 Синтез алгоритма предложения лучших решений пользователю	32
2.3 Обзор возможностей программного-технического обеспечения	33
2.4 Обоснование выбора программного-технического обеспечения	35
2.4.1 Модель обработки данных MapReduce	35
2.4.2 Язык программирования Python	37
2.4.3 Выбор базы данных	38
3 Основные этапы практической разработки программного продукта	40
3.1 Архитектурный проект	40
3.1.1 Диаграмма прецедентов	40
3.1.2 Диаграмма последовательности	43
3.1.3 Диаграмма деятельности	45
3.1.4 Функциональная модель	47
3.2 Проектирование базы данных	53
3.2.1 Инфологическое проектирование	53

3.2.2 Логическое проектирование	61
3.2.3 Физическое проектирование	72
3.3 Разработка алгоритмов сбора данных	78
3.4 Разработка алгоритмов для обработки данных	82
3.5 Разработка алгоритмов для создания модели машинного обучения	84
3.6 Выбор архитектуры	87
3.7 Разработка серверной части	88
3.8 Разработка клиентской части	89
3.9 Фактическое тестирование программного продукта	95
3.10 Анализ достоверности и практической значимости полученных результатов	98
3.10.1 Проверка достоверности полученных решений	98
3.10.2 Практическая значимость полученных решений	99
Заключение	102
Библиографический список	104
Приложение А Программный код	111

## ВВЕДЕНИЕ

Киберспорт – командное или индивидуальное соревнование на основе видеоигр.

Современный мир информационных технологий стал свидетелем стремительного развития киберспорта. Киберспорт уже давно перестал быть просто хобби для геймеров и превратился в масштабную индустрию с многомиллионными доходами. Киберспортивные команды соревнуются на международных турнирах, привлекая внимание миллионов зрителей и спонсоров. В таком динамично развивающемся окружении критическую роль играет эффективное управление и подготовка команд для достижения успеха.

Однако, в киберспорте все еще существуют некоторые проблемы, связанные с обработкой и анализом больших объемов данных. В условиях быстрого темпа развития индустрии, командам сложно удержаться во внимании и добиться успеха без использования передовых информационных технологий. Ключевой вопрос заключается в том, как оптимально использовать огромные объемы данных, чтобы повысить качество подготовки и принимать обоснованные решения во время соревнований.

Цель данной выпускной квалификационной работы – использование алгоритма обработки больших данных MapReduce в разработке программы информационной поддержки киберспортивных команд.

Для создания программного продукта были определены следующие задачи:

- анализ киберспортивной дисциплины «DOTA 2»;
- описание алгоритма обработки больших данных MapReduce;
- проектирование базы данных;
- разработка алгоритмов для сбора данных с использованием MapReduce и для создания модели машинного обучения;
- выбор архитектуры программного продукта;
- обоснование выбора программно-технического обеспечения;

- разработка серверной и клиентской части;
- тестирование и отладка разработанного программного продукта.

В данном контексте, разработка программы информационной поддержки киберспортивных команд является актуальной и важной задачей. Она направлена на сбор и обработку данных о прошедших матчах, анализ показателей игроков и героев, а также предоставление рекомендаций по выбору контрпики и дополнения пика. Это позволит командам обладать более полной информацией, принимать обоснованные решения и повышать свои шансы на успех в соревнованиях.

Более того, разработка программы информационной поддержки киберспортивных команд имеет потенциал для применения не только в профессиональном киберспорте, но и в образовательных учреждениях и любительских командах. Подобная информационная система может помочь улучшить уровень подготовки и обучения игроков, расширить их знания о стратегиях и тактиках, а также повысить интерес к киберспорту в целом.

Таким образом, данное исследование имеет высокую актуальность в контексте быстрого развития киберспорта и необходимости оптимального использования данных для достижения успеха в соревнованиях. Разработка программы информационной поддержки киберспортивных команд представляет собой важный шаг в направлении улучшения управления и подготовки команд, а также способствует развитию киберспорта в целом.

# 1 ОБЩАЯ ХАРАКТЕРИСТИКА ИССЛЕДУЕМОЙ ЗАДАЧИ

## 1.1 Краткие сведения о киберспорте

В современном мире киберспорт перестал быть просто маркетинговым инструментом и стал самостоятельной индустрией, предлагающей уникальные возможности для взаимодействия с молодой аудиторией. Нахождение аудитории, готовой приобрести товар в будущем, становится не менее важным, чем привлечение немедленных потребителей. Киберспорт не только удерживает внимание молодежи в эпоху сокращающейся концентрации внимания, но и создает положительную атмосферу на стадионах, где публика позитивно воспринимает взаимодействие с брендами. Для удовлетворения такого высокого спроса киберспортивные организации быстро адаптируют свои процессы к профессиональному уровню.

Можно заметить, как появляется специализированная инфраструктура, меняется отношение к подготовке игроков, а хаотичная система лиг и турниров превращается в организованный календарь. Команды имеют официальную экипировку, спортсмены берут на себя обязательства перед спонсорами и болельщиками. Результаты этих команд на международной арене играют важную роль в поддержании интереса к профессиональному киберспорту.

В феврале компания NewZoo опубликовала очередной отчет о росте киберспортивного рынка на конец 2020 года.

В феврале компания NewZoo опубликовала отчет о росте киберспортивного рынка на конец 2020 года, который свидетельствует о впечатляющих цифрах.

Мировая киберспортивная экономика сгенерировала доходы в размере 1.1 миллиарда долларов США, что означает рост на 15.7%. Спонсорство и продажа медиаправ составили основную часть доходов суммой в 822.4 миллиона долларов. Продажа билетов принесла 121.7 миллиона долларов, а инвестиции издателей игр в киберспортивное пространство составили 116.3 миллиона долларов. Глобальная киберспортивная аудитория в 2020 году составила 495 миллионов



человек, из которых 272.2 миллиона – активные зрители. Наиболее высокий темп роста киберспорта наблюдается в Юго-восточной Азии (24%), Японии (20.4%) и Латинской Америке (17.9%). Китай остается крупнейшим киберспортивным рынком с выручкой в размере 385.1 миллиона долларов [66].

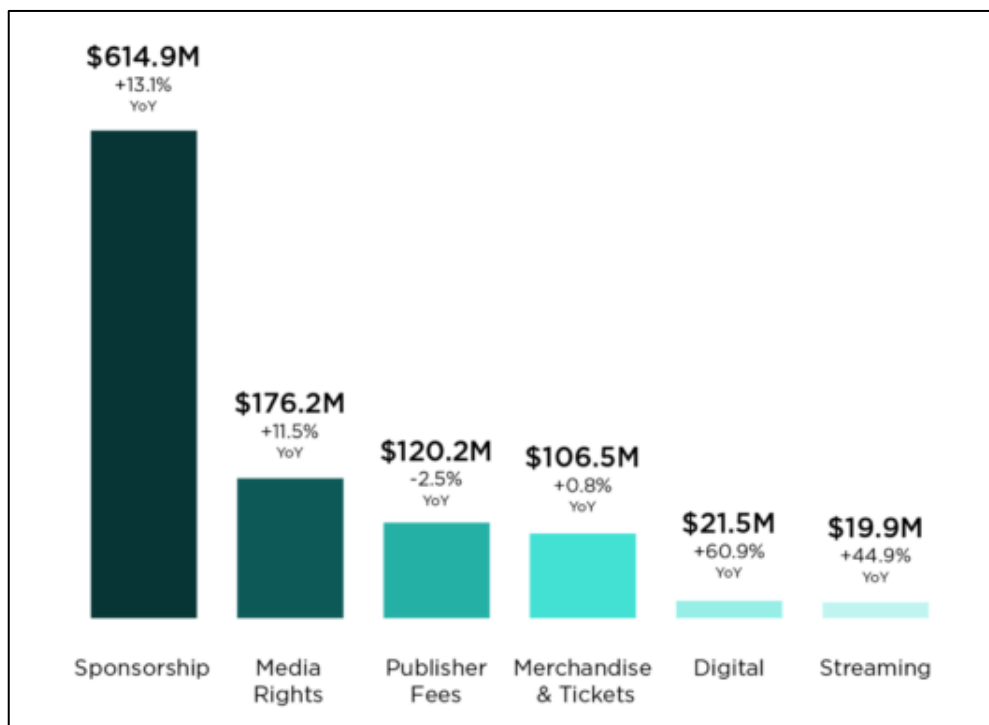


Рисунок 1 – Диаграмма роста киберспортивного рынка на конец 2020 года

С каждым годом появляется все больше стартапов и новых идей, направленных на максимизацию прибыли от киберспорта. Например, на российском рынке киберспорта были совершены две заметные сделки, связанные с приобретением сервисов по сбору игровой статистики и предоставлением игрокам подсказок и обучения в режиме реального времени.

Первым примером сделки такого рода является приобретение сервиса Gosu.ai, который занимается сбором данных об игровой сессии и их обработкой и затем предлагает игрокам подсказки по тактике и оборудованию.

Другим примером действий в данном направлении является киберспортивный стартап Learn 2 Play. Первым проектом компании стала платформа MoreMMR, которая помогает игрокам в Dota 2 выйти на новый уровень с помощью видеоуроков и искусственного интеллекта, анализирующего поведение киберспортсменов во время матча и дающего персональные рекомендации.

Данные сделки примечательны по двум причинам:

- во-первых, сбор и анализ игровых данных позволяют создавать конкурентные преимущества и разрабатывать искусственный интеллект;
- во-вторых, предоставление обучающих сервисов позволяет большему числу игроков переходить от любительского уровня к профессиональному, способствуя росту сегмента.

Киберспорт находится на начальном этапе своего развития, но даже сейчас можно с уверенностью говорить о его перспективности и оценивать его будущие достижения.

## **1.2 Главная киберспортивная дисциплина – DOTA 2**

Dota (Defense of the ancients) 2 – это командная ролевая игра с элементами тактико-стратегических действий. Она возникла как ответвление от игры Warcraft III, однако компания Valve начала разрабатывать собственную игру в 2009 году, и она стала крупнейшей киберспортивной дисциплиной с огромным призовым фондом.

Впервые Dota 2 была представлена широкой публике в августе 2011 года на чемпионате мира The International, организованном компанией Valve. Призовой фонд первого турнира достиг 1 миллион долларов США, что стало рекордом в мире киберспорта. В течение двух лет Dota 2 была доступна только в тестовом режиме, а полноценный релиз состоялся в 2013 году.

В Dota 2 игрокам доступны различные режимы игры, но профессиональные турниры проводятся в основном в режиме Captains Mode. Перед началом матча команды выбирают героев из 120 доступных вариантов. Драфт является важной частью профессиональных матчей, так как правильный выбор героев может существенно повлиять на исход игры.

Игровая карта Dota 2 состоит из симметричных половин, где расположены три линии: нижняя (боттом), центральная (мид) и верхняя (топ). Каждые 30 секунд из зданий на каждой половине карты выходят крипы – базовые существа, за убийство которых игроки получают опыт и золото. Между линиями находится лесная зона, где обитают нейтральные крипы. Опыт позволяет повышать уровень

героя и изучать новые способности, а золото используется для покупки предметов, улучшающих шансы на победу.

После драфта начинается сам матч. Каждый игрок управляет выбранным героем и играет определенную роль в команде. Целью каждой команды является разрушение трона противника, главного здания. Победу также можно достичь, вынудив соперника сдаться из-за экономического превосходства. Преимущество можно получить как в сражениях, так и через правильное использование игровых ресурсов на карте.

Каждой роли в команде соответствует определенная позиция – керри, мидер, оффлейнер, семи-саппорт и фулл-саппорт. Позиции отличаются по доходности.

Керри – самый богатый герой на карте, который наносит основной урон в битвах. Фулл-саппорт – самый бедный персонаж, который помогает и обеспечивает поддержку керри. Мидер также играет важную роль в нанесении урона и занимает центральную линию, часто помогая своим товарищам на боковых линиях. Оффлейнер и семи-саппорт занимают верхнюю или нижнюю линию, в зависимости от стороны, за которую играет команда, и их задача – помешать противнику зарабатывать ресурсы и развивать своего героя. После начального этапа игры все роли объединяются для сражений или захвата объектов на карте.

### **1.3 Рассмотрение существующих методик решения поставленной научной задачи**

#### **1.3.1 Технологии и тенденции работы с Big Data**

В начальных этапах развития технологий больших данных включались различные подходы и инструменты для массово-параллельной обработки неструктурированных данных. Среди них были такие решения, как NoSQL базы данных, модель распределенных вычислений MapReduce и проект Hadoop. Позже к понятию больших данных были отнесены и другие технологии, обеспечивающие сходные возможности по обработке огромных объемов данных, а также аппаратные средства [14].

MapReduce, разработанная Google, представляет модель распределенных параллельных вычислений в компьютерных кластерах. В этой модели приложение разделяется на множество одинаковых задач, выполняемых на узлах кластера, и результаты объединяются в итоговый результат.

NoSQL (Not Only SQL) – общий термин, который охватывает различные нереляционные базы данных и хранилища. Это не определенная технология или продукт. Традиционные реляционные базы данных хорошо работают с быстрыми и однотипными запросами, но при сложных и гибких запросах, характерных для больших данных, использование реляционных баз данных становится неэффективным.

Hadoop – свободный набор инструментов, библиотек и фреймворк для разработки и выполнения распределенных программ, предназначенных для работы на кластерах из сотен и тысяч узлов. Он считается одной из основных технологий в области больших данных.

R – язык программирования, используемый для статистической обработки данных и создания графиков. Он широко применяется для анализа данных и является фактическим стандартом для статистических программ.

Также аппаратные решения играют важную роль в области обработки больших данных. Компании, такие как Teradata и EMC, предлагают готовые к использованию аппаратно-программные комплексы, включающие кластеры серверов и программное обеспечение для массово-параллельной обработки данных. В эту категорию также входят аппаратные решения для аналитической обработки в оперативной памяти, такие как Hana от SAP и Exalytics от Oracle, хотя такая обработка изначально не является массово-параллельной, а объем оперативной памяти одного узла ограничен несколькими терабайтами [3].

### 1.3.2 Методы и техники анализа больших данных

Консалтинговая компания McKinsey выделяет 11 методов и техник анализа, применимых к большим данным:

- Data Mining (добыча данных, интеллектуальный анализ данных, глубокий анализ данных) – это совокупность методов, которые позволяют

обнаруживать ранее неизвестные, нетривиальные и практически полезные знания в данных, включает методы, такие как обучение ассоциативным правилам, классификация, кластерный анализ, регрессионный анализ, обнаружение и анализ отклонений и другие;

- краудсорсинг – это метод, при котором широкий и неопределенный круг людей классифицирует и обогащает данные, не вступая в трудовые отношения, техника позволяет обрабатывать большие объемы данных из множества источников, и количество участников не ограничено;

- смешение и интеграция данных (data fusion and integration) – это набор техник, позволяющих интегрировать разнородные данные из различных источников для проведения глубинного анализа, примеры включают цифровую обработку сигналов, обработку естественного языка и тоновый анализ;

- прогнозирование – аналитики определяют параметры системы заранее и затем проверяют поведение объекта на основе больших объемов информации, помогает предсказывать будущие события и тенденции на основе данных;

- машинное обучение, включая обучение с учителем и без учителя – это использование моделей, построенных на основе статистического анализа или машинного обучения, для получения комплексных прогнозов на основе базовых моделей, позволяет анализировать и обрабатывать большие объемы неструктурированных данных и использовать их для автоматического обучения;

- искусственные нейронные сети, сетевой анализ, оптимизация, включая генетические алгоритмы – это методы, используемые для оптимизации и моделирования данных, они позволяют находить оптимальные решения и прогнозировать результаты на основе анализа больших объемов данных;

- глубинный анализ, классификация данных – это методы, которые изначально были разработаны для работы с структурированной информацией в небольших объемах данных, однако с применением усовершенствованных математических алгоритмов они также могут быть применены к большим данным;

- анализ сетевой активности – это методика использования big data для исследования социальных сетей, взаимосвязей между владельцами аккаунтов, группами и сообществами, это позволяет создавать целевые аудитории на основе интересов, геолокации, возраста и других метрик;
- имитационное моделирование (simulation) – это метод, позволяющий создавать модели, которые описывают процессы, происходящие в реальном мире, он является разновидностью экспериментальных исследований;
- пространственный анализ (spatial analysis) – это класс методов, использующих топологическую, геометрическую и географическую информацию, извлекаемую из данных, позволяет анализировать пространственные взаимосвязи и паттерны в больших объемах данных;
- статистический анализ – это анализ временных рядов, проведение А/В-тестирования и других методов для выявления взаимосвязей и влияния факторов на целевые показатели, эти методы позволяют выбрать несколько элементов из массива данных и сравнить их до и после изменений для определения влияния различных факторов;
- визуализация аналитических данных – это представление информации в виде графиков, диаграмм и с использованием интерактивных возможностей и анимации, играет важную роль в анализе больших данных, позволяя наглядно представить результаты и использовать их в качестве исходных данных для дальнейшего анализа [60].

### 1.3.3 Аналоги разрабатываемого проекта

В данной главе представлен обзор аналогов разрабатываемого проекта. Важно отметить, что данные аналоги не являются прямыми соответствиями разрабатываемого решения, так как настоящие аналоги разрабатываются специально для команды и не представлены в открытом доступе. Однако, рассмотрение этих аналогов позволяет получить представление о схожих решениях и функционалах, которые могут быть применимы и полезны для проекта.

Feedless, инновационная программа-компаньон, предоставляет игрокам настоящую помощь и полезные советы на английском языке во время игры Dota 2.

Ее функционал включает оповещения о соперниках, покинувших линию, предсказание времени появления руны, напоминания о необходимости стакать крипов, а также указания на предстоящий спавн вардов. Большинство функций доступны в стандартной версии, но улучшенная версия Feedless предлагает дополнительные возможности, такие как напоминания о необходимости носить свиток телепортации и предупреждения о долгом хранении вардов в инвентаре. Кроме того, программа помогает в выборе предметов, информирует о возможности ганка и предупреждает о возможной смертельной атаке на игрока.

Принцип работы Feedless заключается в том, что программа делает скриншоты экрана во время игры, отправляет их на сервер для анализа и предоставляет игрокам соответствующие советы. Влияние программы на производительность не существенно, хотя на очень слабых компьютерах может возникнуть проблема с переходом в режим "окно на весь экран", так как Dota 2 может работать немного медленнее в этом режиме.

Feedless активируется во время пика игры. В окне отображаются рекомендуемые герои, сочетающиеся с союзниками и имеющие преимущество над противниками. Игрок может указать предпочитаемую линию и роль, чтобы получить более точные советы. В других разделах можно просмотреть ожидаемое размещение героев противников по линиям, информацию о героях, на которых играли все участники матча, и оценить силу различных комбинаций команд на разных этапах игры. Интерфейс программы может показаться непонятным, и разобраться в нем за ограниченное время во время пика может быть сложно. Программе явно не хватает предварительного представления, объяснений до начала игры или даже руководства на сайте.

После начала игры окно программы исчезает, и Feedless начинает вести диалог с игроком. Она сообщает, когда соперники вышли из поля зрения на других линиях, предупреждает о рунах и стаках, рекомендует цели для ганков,

напоминает о необходимости носить с собой свиток телепортации и предупреждает о скором спавне вардов. В общем, программа предоставляет игроку разнообразную информацию об игре. Она также помогает игрокам обратить внимание на важные аспекты игры.

Gosu.ai, находящийся в стадии альфа-тестирования, является онлайн-сервисом для анализа матчей. Он оценивает выбор предметов игроков в начале игры, информирует об их вкладе в победу, отображает количество пропущенных крипов и время, потраченное на использование мыши вместо горячих клавиш. Взаимодействие с программой осуществляется через бота Steam, который необходимо добавить в список друзей.

В разделе, посвященном фарму, предоставляется информация о количестве пропущенных крипов в первые 10 минут игры и причинах этого – например, сколько раз игрок не смог убить крипов, сколько раз его отвлекал оффлейнер или сколько крипов было убито башней. Понимание, где игрок теряет золото на раннем этапе игры, полезно, но только практика поможет стать сильнее.

Pvgnа – ресурс с англоязычными видеогайдами, разделенными на ветки по различным темам в Dota 2. Ветки включают такие общие темы, как повышение MMR, игра на керри, саппорте, оффлейнере и мидере. Также имеется раздел, посвященный базовым механикам и концепциям. Ветки содержат гайды по более конкретным темам, начиная с основ и продвигаясь к более узким руководствам. Каждый гайд разделен на небольшие видеоролики примерно по 15 минут. Некоторые видео не относятся к основным темам.

На сайте руководства создаются как профессионалами, например, Расмусом "Chessie" Бломдиным, Эндрю "Jenkins" Дженкинсом, Никласом "Wagamama" Хогстромом и даже обычными игроками с разным уровнем MMR. Авторы могут представить одну и ту же тему по-разному: один будет делиться своим опытом и знаниями, другой сослагательно будет описывать, как играют профессионалы, а третий предложит теоретический подход. Поэтому на одних героев можно найти несколько гайдов. При поиске руководств можно указать, ищите ли гайд, обзор игры или анализ матча профессионального игрока.



Ресурс имеет мобильное приложение, которое отличается от веб-сайта – оно не имеет общих веток. На главной странице приложения размещены видео, которые Pvnna считает актуальными для конкретного пользователя, например, гайды по героям, которыми недавно играли. Лучше всего использовать мобильную версию как дополнение к веб-сайту – если, например, не успели досмотреть видео дома, можно продолжить его просмотр в пути. Однако здесь не хватает синхронизации между веб-сайтом и приложением, а также функций, таких как "посмотреть позже на смартфоне" или общей истории просмотра. Видео нельзя сохранять, поэтому требуется постоянное подключение к интернету.

Oracle – бесплатный сервис для анализа игроков и матчей в Dota 2, также известный как "тренер Dota 2 с искусственным интеллектом". Программа оценивает игрока по шести пунктам: мобильность, фарм, командные сражения, выживаемость, предметы и обзор, а затем сравнивает результаты с средними показателями на его уровне MMR. Каждый пункт разбит на подпункты, сопровождаемые графиками и пояснениями, показывающими их важность и причины, а также насколько лучше игроки с более высоким уровнем MMR справляются с этими пунктами. Например, параметр выживаемости включает в себя такие показатели, как количество использований Magic Wand в минуту, количество смертей за матч, время, проведенное мертвым, частоту участия в командных сражениях и процент золота, потерянного из-за смертей. Также можно указать игровую роль, и тогда характеристики в пунктах будут изменены – программа измеряет керри и саппортов по разным показателям.

Оценка матчей в Oracle производится менее подробно. Она сравнивает, насколько обе команды хорошо проявили себя по шести параметрам и оценивает, насколько каждый игрок выполнил свою роль. Однако в оценке ролей не хватает пояснений – почему именно так и что можно было сделать лучше.

MoreMMR – обучающий сайт с видеороликами и заданиями. Придумал портал один из основателей «Нетологии-групп», компании, которая занимается онлайн-образованием.

В каждом видео рассказывается о какой-то одной теме – какие бывают роли, что такое сплит-пуш, как правильно выбрать героя на линию, как играть на том или ином персонаже. В ранних пунктах даётся общая информация, которая может быть полезна разве что новичкам.

GameLear – это англоязычный сборник видеогайдов с чёткой структурой. Ресурс разделён на две части: общие гайды и руководства для героев. В общих гайдах рассматриваются механика, роли, психология Dota 2, а также анализ реплеев высокоуровневых игроков. Раздел с героями предлагает 101 руководство, обычно продолжительностью от полутора до двух часов. Самый длинный гайд, посвящённый герою Arc Warden, составляет почти 6 часов. В отличие от Pvnа, здесь гайды не повторяются, обычно для каждого персонажа имеется одно руководство, хотя для некоторых героев есть видео для разных ролей.

Каждый гайд разбит на тематические части, выбранные автором, поэтому качество и полезность гайдов могут различаться. Один автор может уделять особое внимание ранней стадии игры, другой подробно объяснять, как принимать правильные решения на герое, а третий посвятить полчаса разбору ситуаций, когда следует выбирать персонажа. Некоторые гайды могут быть незавершёнными и состоять только из введения и нескольких разделов.

GameLear особенно полезен для тех, кто учится играть на новых героях, но для опытных игроков информации может быть недостаточно. Основатель ресурса считает, что он наиболее полезен для игроков с низким уровнем игры.

Overwolf – израильская технологическая платформа, которая позволяет сторонним разработчикам создавать, распространять и монетизировать внутриигровые приложения и моды. Overwolf DotaPlus – специальное дополнение, созданное для Overwolf.

Функциональность Overwolf DotaPlus включает просмотр статистики пользователей и их персонажей непосредственно в процессе игры, также в приложении есть инструмент для просмотра группы, в которой сейчас участвует тот или иной геймер.

В приложении Overwolf Dota Plus:

- можно увидеть информацию о результатах пользователя (число матчей и какова в них доля побед);
- данные геймера за предыдущие три месяца (самых часто выбираемых персонажей);
- показатели GPM (количество зарабатываемого в минуту золота) и XPM (объем опыта в единицу времени);
- какой у этого персонажа показатель помощи, смерти и убийств и его GPM и XPM.

В данной главе были рассмотрены несколько аналогов программного обеспечения. Изучение данных аналогов позволяет получить общее представление о схожих решениях и функционалах, которые могут служить источником вдохновения и полезной информации для проекта. Настоящие аналоги разрабатываются с учетом уникальных требований и потребностей команды, поэтому точный функционал и преимущества остаются недоступными для описания в данном обзоре.

## 2 АЛГОРИТМИЧЕСКОЕ И ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ РЕШЕНИЯ ЗАДАЧИ

### 2.1 Синтез алгоритма сбора данных

#### 2.1.1 Теоретическая информация об алгоритме сбора данных

Веб-скрейпинг – это инновационная технология, позволяющая преобразовывать HTML-страницы в форматы, более удобные для дальнейшего использования. Эта технология является частью широкого концепта web mining (WM), который фокусируется на извлечении ценной информации из различных источников данных, доступных в Интернете. Уже с середины 90-х годов WM привлекает внимание исследователей, и существовали два подхода к этой области: процессный и дата-центричный. Однако второй подход получил большее распространение. Сегодня WM включает методы автоматического систематического обхода Интернета с целью сбора ценных данных о компаниях, людях и других объектах для принятия информированных решений.

WM сосредоточен на работе с тремя типами информации, каждый из которых имеет свой вид WM:

- данные о действиях пользователей, которые включают журналы серверов и информацию о взаимодействии с браузером, это известно как Web Usage Mining (WUM) или нагрузочный WM;
- веб-графы, описывающие связи между страницами веба, они представляют собой вершины, представляющие веб-страницы, и дуги, представляющие гиперссылки между ними, в области Web Content Mining (WCM) проводятся активные исследования, и основные вызовы связаны с разнообразием данных веба и их низкой структурированностью, что затрудняет извлечение нужной информации;
- социальные сети и сообщества: в сети Интернет существует множество социальных платформ, где люди активно обмениваются информацией, создают профили и устанавливают связи друг с другом, Social Network Analysis (SNA) или социальный WM занимается извлечением и анализом данных из этих

социальных сетей, это может включать анализ социальных графов, выявление влиятельных пользователей, определение групп и сообществ, анализ тематик обсуждений и даже предсказание поведения пользователей на основе их социальной сети.

Кроме того, WCM включает решение специфических задач, таких как:

- извлечение структурированных данных из веб-страниц с использованием методов машинного обучения и нейронных сетей;
- унификация форматов представления данных и их интеграция из различных источников;
- оценка продуктов и услуг на основе отзывов, размещенных на форумах, блогах и чатах.

В WCM широко применяется процедура перевода данных из формы, предназначенной для чтения человеком, в форму, удобную для обработки компьютером. Изначально такая процедура называлась *data scraping*, но со временем она претерпела изменения и получила название *web scraping*.

Однако полностью автоматизированная генерация данных остается на экспериментальном уровне, и высшим достижением веб-скрейпинга являются анализаторы веб-страниц, основанные на системах компьютерного зрения и машинного обучения, включающих элементы искусственного интеллекта.

Веб-скрейпинг работает с тремя типами информации: структурированными, неструктурированными и квазиструктурированными. Для обработки структурированных данных обычно применяются служебные процедуры, включающие обход страниц, генерацию и исполнение упаковщика, а затем переход к анализу содержимого страницы.

Сложнее обстоит дело с квазиструктурированными данными, как, например, граф посещений сайта. Для работы с такими данными были разработаны специальные языки, включая NEXIR ELOG, которые предназначены для программирования упаковщиков. С помощью этих языков описывается процедура выделения данных, которая завершается созданием объектной модели данных (ОЕМ).

В рамках неструктурированного извлечения данных (DM) используются различные операции, которые облегчают восприятие больших объемов текстов пользователем. Среди них выделяются следующие типы операций:

- отслеживание тематики: оценка области интересов пользователя;
- свертка: создание резюме документов;
- ранжирование: упорядочение документов и их распределение по заранее определенным категориям;
- кластеризация: объединение схожих документов в группы;
- визуализация данных: решение проблемы коммуникации пользователя с данными;

На сегодняшний день около дюжины компаний разрабатывают инструменты для неструктурированного извлечения данных в виде загружаемых коммерческих программ, свободно-распространяемых программ и облачных сервисов. Несмотря на разнообразие таких инструментов, они имеют некоторую общую структуру.

Так, любой инструмент для веб-скрейпинга включает следующие элементы:

- краулинг: процесс обхода веб-страниц для сбора информации;
- парсинг: синтаксический анализ полученных данных для их дальнейшей обработки;
- хранение информации: сохранение собранных данных для последующего доступа и использования;

Поисковый робот или краулер представляет собой программу, которая осуществляет краулинг и собирает информацию о веб-страницах для хранения в базе данных. Краулинг позволяет автоматизировать сбор различных видов информации, включая, например, адреса электронной почты.

Работа поискового робота основана на тех же принципах, что и у веб-браузера. Робот посещает веб-сайты, анализирует содержимое страниц и передает их в базу данных поисковой системы. Затем он следует по ссылкам, переходя на

другие ресурсы и повторяя этот алгоритм действий. Результатом такого поиска является последовательное перебирание веб-ресурсов. При доступе к веб-ресурсу робот находит файл robots.txt, который содержит правила для дальнейших действий робота в соответствии с указанными правилами.

При использовании технологии краулинга необходимо задавать следующие характеристики:

- глубина: определяет количество кликов от стартовой страницы, на которое необходимо распространиться;
- количество параллельных загрузок страниц;
- пауза между загрузками страниц: используется для снижения нагрузки на сервер, значение этого параметра часто определено в файле robots.txt, который задан владельцем ресурса;

Важно, чтобы краулер не создавал слишком большую нагрузку на сервер сайта, иначе такая активность может рассматриваться как кибератака и привести к блокировке как самого сайта, так и краулера.

Библиотека Scrapy – это прикладная среда для сканирования веб-сайтов и извлечения структурированных данных. Она может быть использована для различных полезных приложений, таких как анализ данных, обработка информации или архивирование данных.

Несмотря на то, что Scrapy изначально был разработан для сканирования веб-страниц, его также можно использовать для извлечения данных через API (например, Amazon Associates Web Services) или в качестве универсального веб-сканера [69].

Scrapy предоставляет множество мощных функций для удобного и эффективного очищения данных, таких как:

- встроенная поддержка выбора и извлечения данных из источников HTML/XML с использованием расширенных селекторов CSS и выражений XPath, а также вспомогательных методов с использованием регулярных выражений;

- интерактивная оболочка консоли IPython для обработки выражений CSS и XPath;
- встроенная поддержка генерации и экспорта данных в нескольких форматах (JSON, CSV, XML) и их хранения в различных хранилищах (FTP, S3, локальная файловая система);
- надежная поддержка кодирования и автоопределения для работы с внешними, нестандартными и некорректно объявленными кодировками;
- поддержка расширяемости, которая позволяет вам добавлять свои собственные функции, используя сигналы и четко определенное API (middleware, расширения и конвейеры);
- широкий спектр встроенных расширений и промежуточного программного обеспечения для обработки:
  - обработка cookies и сессий;
  - функции HTTP, такие как сжатие, аутентификация и кэширование;
  - замена пользовательского агента;
  - учет файла robots.txt;
  - ограничение глубины сканирования.

Процесс веб-скрейпинга состоит из двух этапов:

- систематический поиск и загрузка веб-страниц;
- извлечение данных с веб-страниц.

Создание поискового робота с нуля может быть выполнено с использованием различных модулей и библиотек, предоставляемых выбранным языком программирования. Однако, по мере развития программы, это может вызвать некоторые проблемы. Например, может потребоваться преобразовать извлеченные данные в форматы CSV, XML или JSON. Также, можно столкнуться с веб-сайтами, требующими специальных настроек и моделей доступа.



Поэтому рекомендуется сразу разработать робота на основе библиотеки, которая решает эти потенциальные проблемы. В этом случае Python и Scrapy являются отличным выбором.

Scrapy является одной из наиболее популярных и эффективных библиотек Python для извлечения данных с веб-страниц. Она включает в себя множество функциональных возможностей, что позволяет избежать необходимости реализации этих функций самостоятельно. Scrapy облегчает и ускоряет создание "веб-паука" без лишних трудозатрат.

Пакет Scrapy, как и многие другие пакеты Python, доступен в PyPI (Python Package Index), который является репозиторием, поддерживаемым сообществом, и содержит множество пакетов Python.

Поисковые роботы позволяют извлекать информацию о различных продуктах, получать большие объемы текстовых или числовых данных, а также извлекать данные с веб-сайтов без использования официального API и многое другое.

Файл robots.txt является текстовым файлом, расположенным в корневой директории сайта, и содержит специальные инструкции для поисковых роботов. Эти инструкции определяют параметры индексирования сайта. В файле robots.txt можно указать запрет индексации определенных разделов или страниц сайта, а также рекомендовать поисковым роботам соблюдать определенный временной интервал между запросами к серверу или блокировать неважные изображения и скрипты. После ознакомления с файлом, роботы посещают главную страницу сайта и следуют ссылкам, двигаясь вглубь [24].

Индексация подразумевает анализ и добавление страниц сайта в базу данных поисковых систем, что позволяет сайту появляться в результатах поисковых запросов.

При работе с файлом robots.txt, роботы в первую очередь проверяют наличие записей, начинающихся с User-agent. Эти записи содержат инструкции для конкретного робота. Однако некоторые поисковые роботы не учитывают общие

запрещающие правила в файле robots.txt, поэтому инструкции для таких роботов требуется указывать отдельно.

После указания user-agent следуют команды, которые запрещают или разрешают доступ к определенным разделам сайта. Для запрета доступа используется директива Disallow, а для разрешения доступа - директива Allow, с параметрами, которые указывают URL раздела или страницы сайта.

Обычно директива Disallow используется для запрета доступа к служебным разделам, конфиденциальной информации и удаления дублирующихся страниц. Под дублирующимися страницами понимаются страницы, которые доступны по разным адресам.

Директива Allow в файле robots.txt используется в основном для указания роботам поисковых систем страниц, которые должны быть индексированы. Она помогает правильно проиндексировать сайт. По умолчанию все страницы сайта считаются разрешенными для индексации, поэтому директива Allow реже используется по сравнению с директивой Disallow.

Однако в контексте веб-скрапинга, важной и полезной директивой в файле robots.txt является crawl-delay. С ее помощью владелец сайта может указать минимальный интервал времени между загрузкой одной страницы и следующей. Это полезно, когда сервер перегружен и не успевает обрабатывать запросы на загрузку страниц.

Также в файле robots.txt существуют директивы host и sitemap. Директива host указывает основное зеркало сайта, а директива sitemap указывает URL карты сайта, которая представляет структуру сайта и расположение его контента. Зеркало сайта является дубликатом сайта, полным или частичным [40].

Парсер – это программа, которая автоматизирует обработку информации, полученной с веб-сайта. Он анализирует и преобразует текст, выделяя определенные фрагменты или удаляя ненужную информацию. Парсер быстро и точно отделяет нужную информацию от лишней в HTML-документе и упаковывает результат в определенном формате [39].

Существует несколько методов извлечения данных:

- анализ DOM-дерева HTML страницы: в этом подходе данные извлекаются на основе структуры DOM-дерева HTML/XML документа, можно получать данные, обращаясь к атрибутам элементов дерева или спускаясь по дереву, однако этот метод требует привязки к движку и может потерять доступ к элементам при изменении их расположения;
- парсинг строк: этот метод применяется для данных с фиксированным форматом, где данные извлекаются путем разбора отдельных строк;
- использование регулярных выражений: регулярные выражения часто используются для решения простых задач или написания специальных процедур для извлечения данных;
- XML-парсинг: этот метод рассматривает HTML как XML-данные, поскольку HTML редко является валидным XML, библиотеки, реализующие этот подход, часто выполняют преобразование HTML в XML перед парсингом данных;
- визуальный подход: этот подход позволяет пользователю настраивать систему для извлечения данных различной сложности и вложенности без программирования или использования API, однако такой подход находится на начальной стадии развития [11].

Каждый из этих методов имеет свои преимущества и ограничения, и выбор подходящего метода зависит от конкретных требований и характеристик извлекаемых данных.

При парсинге HTML данных могут возникать следующие проблемы:

- использование JavaScript, AJAX и асинхронных загрузок: если данные на странице загружаются или обновляются с использованием JavaScript, AJAX или асинхронных запросов, это усложняет написание парсеров, так как требуется обрабатывать динамически изменяющийся контент или выполнение JavaScript кода для получения полных данных;
- различные движки для рендеринга HTML: различные браузеры и движки для рендеринга HTML (например, WebKit, Gecko) могут обрабатывать

HTML по-разному, что может приводить к различиям в структуре DOM дерева, это может усложнить разработку универсальных парсеров, которые должны работать на разных платформах;

– большие объемы данных: парсинг больших объемов данных может потребовать использования распределенных парсеров, которые работают параллельно для обработки данных, это может повлечь дополнительные затраты на синхронизацию, управление ресурсами и обработку ошибок.

Для решения этих проблем могут использоваться различные техники, такие как использование библиотек для работы с JavaScript (например, Selenium), анализ событий загрузки страницы, асинхронное выполнение запросов, а также адаптация парсера под конкретные движки и структуру данных.

#### 2.1.2 Автоматизированный алгоритм для сбора данных

В развитии Интернета информационные ресурсы играют все более важную роль, и существует потребность в эффективном отборе и сборе информации из сети Интернет. Для автоматизации этого процесса разработаны интеллектуальные помощники, которые используются для сбора информации из различных веб-ресурсов в соответствии с запросами пользователей.

Сбор информации в Интернете является трудоемким и рутинным процессом, который требует много времени. Парсеры, или веб-скрейперы, позволяют автоматизировать эту работу путем перебора большого количества веб-ресурсов за короткое время. Они позволяют собирать нужную информацию с веб-страниц, обрабатывать и анализировать ее.

Поисковые системы являются одним из наиболее активно используемых инструментов для парсинга информации. Роботы поисковых систем собирают информацию в широких целях, включая автоматическую проверку уникальности текстовой информации. Они могут быстро сравнивать содержимое множества веб-страниц с предложенным текстом для определения его уникальности.

Использование парсеров и автоматических инструментов для сбора информации позволяет существенно упростить и ускорить процесс получения необходимых данных из Интернета.

Первыми в работу вступят скрапперы для сбора данных о командах, игроках, турнирах, героях, предметах и сыгранных матчах. В общей сложности, каждый скраппер выглядит похоже, в них присутствует разница лишь в первой получаемой ими ссылки, а также в собираемых данных.

Рассмотрим алгоритм работы скраппера для сбора данных о командах и актуальных составах (рис. 2).

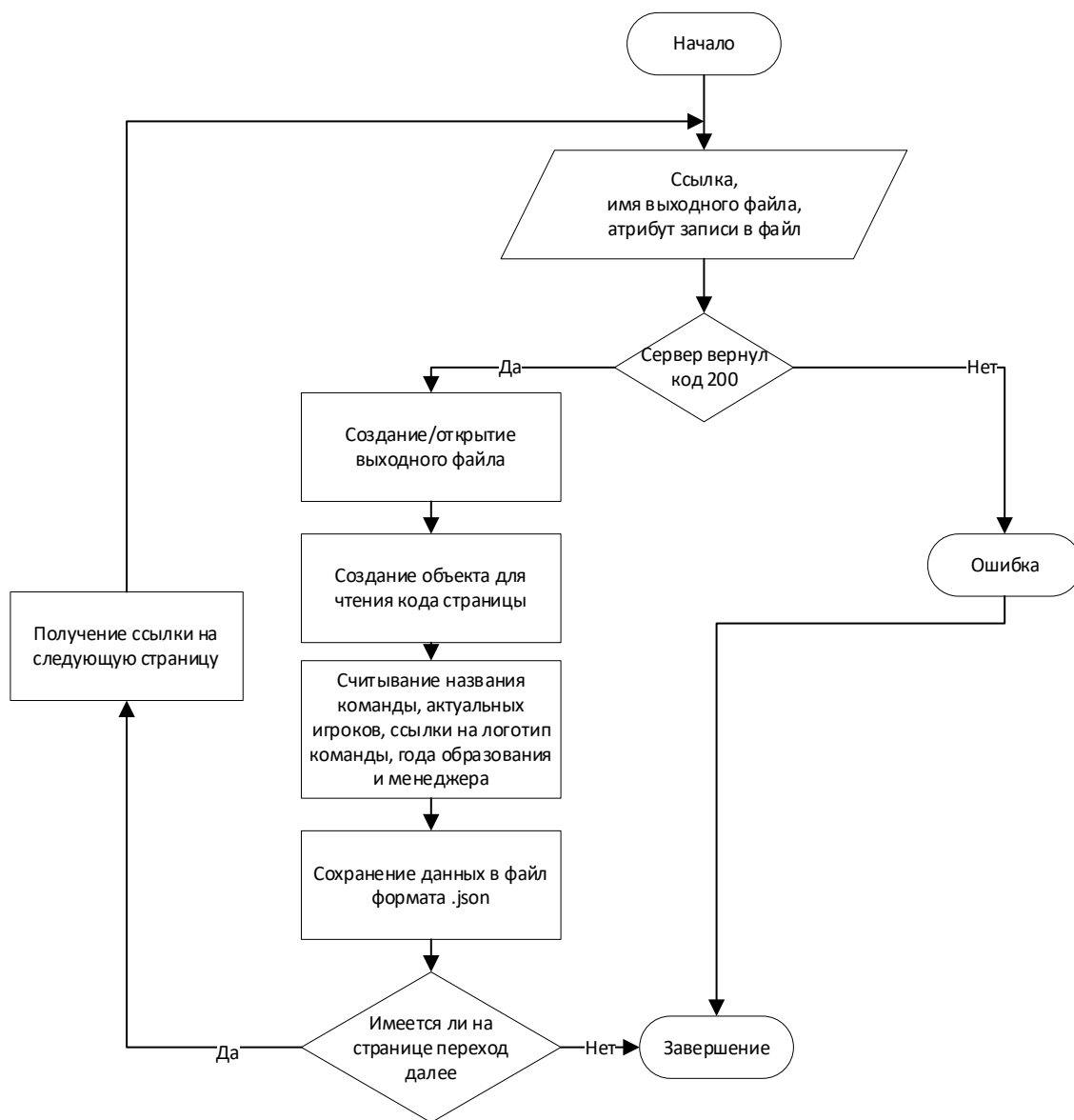


Рисунок 2 – Алгоритм получения данных о командах

При запуске сначала отправляется запрос на страницу по ссылке, указанной при вызове, также для корректного запроса передаются http-заголовки. Заголовки HTTP позволяют клиенту и серверу отправлять дополнительную

информацию с HTTP запросом или ответом. Если запрос не вернет «код ответа 200», то работа будет завершена. Иначе, будет продолжена работа на данной странице.

Будет создан или открыт уже имеющийся выходной файл, после создастся объект для работы с разметкой текущей страницы, из которого уже и будет браться вся информация. Полученные данные записываются в файл формата .json. После чего проверяем имеется ли на странице переход на следующую страницу, если имеется, то переходим на следующую страницу, иначе завершаем работу данного модуля.

Следующий этап – очистка собранных данных. В связи с тем, что собранные данные о прошедших матчах могут включать неактуальную информацию о составах команд, то прежде, чем производить загрузку базы данных, необходимо вычлениить из списка те записи, которые устарели.

В данном случае поможет модель обработки данных MapReduce. Будем использовать её для агрегирования данных по командам. В качестве ключа можем использовать название команды, а в качестве значения – игроков команды. Функция Mapper будет принимать на вход данные в формате «команда – игроки» и возвращать набор промежуточных пар «команда – игрок». Функция Reducer будет получать набор промежуточных пар с одинаковыми ключами (названиями команд) и складывать активных игроков для каждой команды. В результате получим набор итоговых пар «команда – игроки» (рис. 3).

После очистки, оставшиеся данные записываются в файл.

Последний этап – заполнение базы данных. Сначала необходимо создать подключение. Так как, база данных находится на выделенном сервере, то и время подключения, и время отклика зависит от стабильности интернет-соединения как у клиента, так и у сервера базы данных.

Дальше последовательно загружаются данные из подготовленных файлов, в соответственные коллекции (рис. 4).

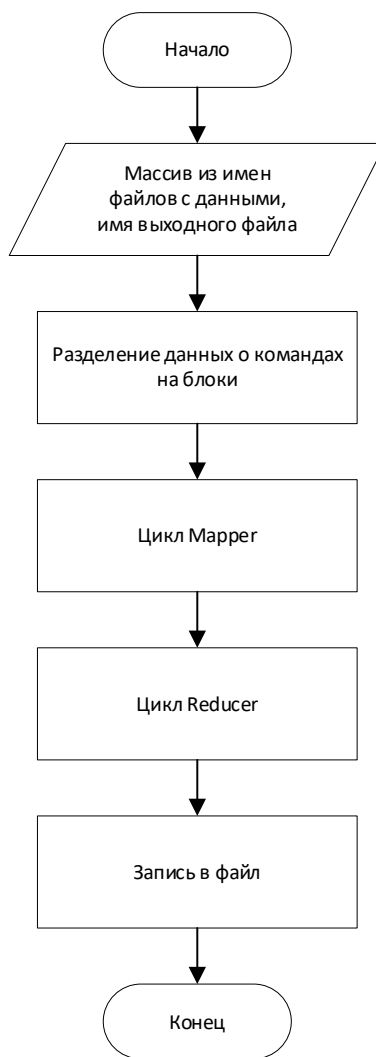


Рисунок 3 – Алгоритм отчистки данных



Рисунок 4 – Алгоритм записи в базу данных

В результате проведенной работы по синтезу алгоритма сбора данных были разработаны эффективные и надежные схемы работы алгоритмов. Эти схемы представляют собой систематический подход к сбору данных, учитывающий особенности и требования работы.

## 2.2 Синтез алгоритма предложения лучших решений пользователю

В реализованном программном продукте участвует модель машинного обучения. В первом случае она реализует предложение для выбора героя против определенного героя, выбранного командой соперника. Во втором предлагает выбор героя для определенной игровой роли на основе выбора нескольких героев.

Так как все данные хранятся удаленно, на сервере, то для начала необходимо создать скрипт, который будет обращаться к необходимой таблице в базе данных, считывать данные из таблицы и переносить их в датасет-файл.

Алгоритм работы представлен в виде блок-схемы (рис. 5).

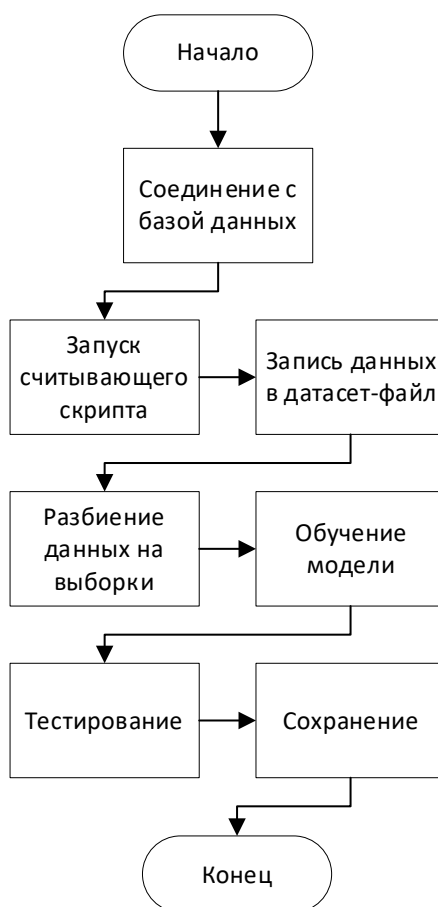


Рисунок 5 – Алгоритм обучения модели



Считав данные приступаем к разбиению данных на выборки, необходимые для обучения модели и ее последующего тестирования. Выполнив данный этап необходимо приступать к обучению модели. Этот процесс также включает в себя проверку модели на тестовой выборке. Обучив и протестировав, теперь ее необходимо сохранить, для дальнейшего использования.

### 2.3 Обзор возможностей профильного программного обеспечения

Visual Studio – полнофункциональная IDE от Microsoft, которая во многом сопоставима с Eclipse. Доступная на Windows и Mac OS, Visual Studio представлена как в бесплатном (Community), так и в платном (Professional и Enterprise) вариантах. Visual Studio позволяет разрабатывать приложения для разных платформ и предоставляет свой собственный набор расширений.

Python Tools for Visual Studio (PTVS) позволяет создавать код на Python в Visual Studio и включает в себя Intellisense для Python, отладку и другие инструменты.

Таблица 1 – Сравнение версий Visual Studio

Характеристики	Visual Studio Community	Visual Studio Professional
1	2	3
Поддержка языка Python	+	+
Поддержка TFS	-	+
Модульное тестирование	+	+
Тесты производительности	-	+
Кроссплатформенность	+	+
Система управления версиями	+	+
Рефакторинг	+	+
Интерактивная оболочка REPL	+	+

1	2	3
Возможности удаленной разработки	-	+
Поддержка баз данных и SQL	+	+

Ключевыми преимуществами Visual Studio являются:

- редактор кода, содержащий инструменты для работы с языками, подсвечивание синтаксиса, ошибок, типов данных и прочего;
- автоматизация сборки;
- дебаггер, или отладчик, помогает детально изучить код и найти в нем ошибки, отсутствие подобного средства усложняет написание больших мобильных или веб-приложений, минимизирует риск ошибки при скоростной печати символов;
- IntelliSense, это инструмент для исправления ошибок разработчика, также он предлагает ряд вариантов, а также в отдельных случаях может сгенерировать фрагменты кода;
- CodeLens, утилита для нахождения ошибок в коде разработчика;
- универсальность языков;
- меньше письма – больше результата, работая с другими средствами для разработки, необходимо писать многие вещи вручную, которые в VS добавляются благодаря интеллектуальной системе;

Пример создания проекта в Visual Studio на языке Python представлен на рисунке 6.

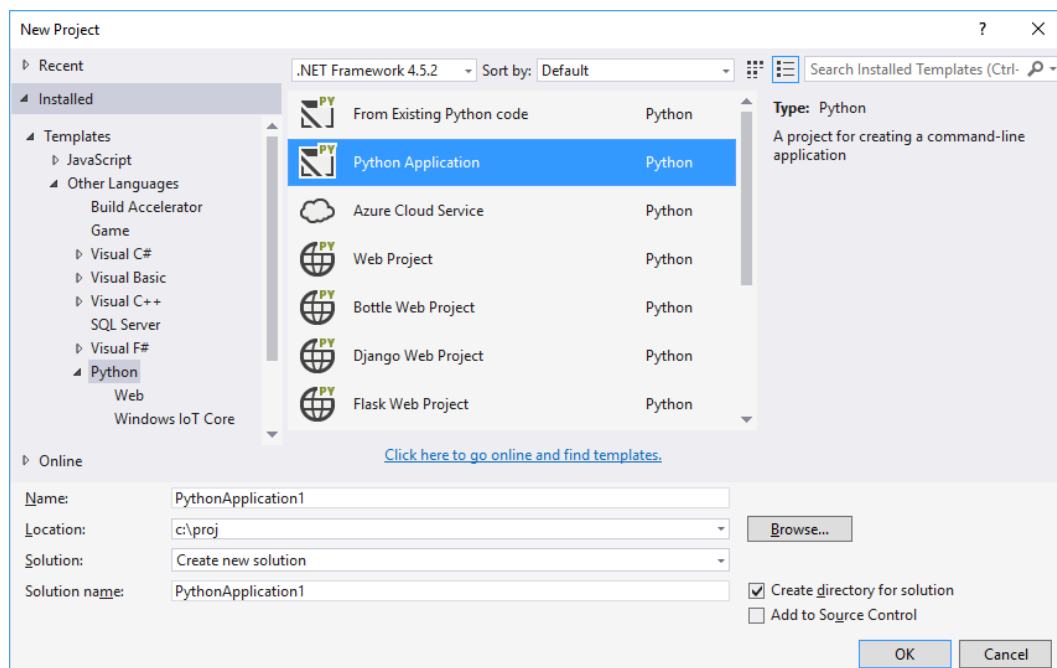


Рисунок 6 – Окно создания проекта в Visual Studio

## 2.4 Обоснование выбора программно-технического обеспечения

### 2.4.1 Модель обработки данных MapReduce

После проведения исследования различных методов обработки Big Data была выбрана модель MapReduce.

MapReduce – это модель распределенной обработки данных, предложенная компанией Google для работы с большими объемами данных на компьютерных кластерах.

В рамках модели MapReduce данные структурируются в виде записей и проходят через три стадии обработки.

Стадия Map, где данные предварительно обрабатываются при помощи пользовательской функции `map()`. Здесь происходит предварительная обработка и фильтрация данных. Функция `map()` применяется к каждой входной записи и генерирует набор пар ключ-значение. Этот набор может содержать одну запись, не содержать ничего или содержать несколько пар ключ-значение. Пользователь определяет, какие данные будут использованы в качестве ключа и значения. Ключ играет важную роль, поскольку данные с одинаковыми ключами будут переданы в один экземпляр функции `reduce()`.

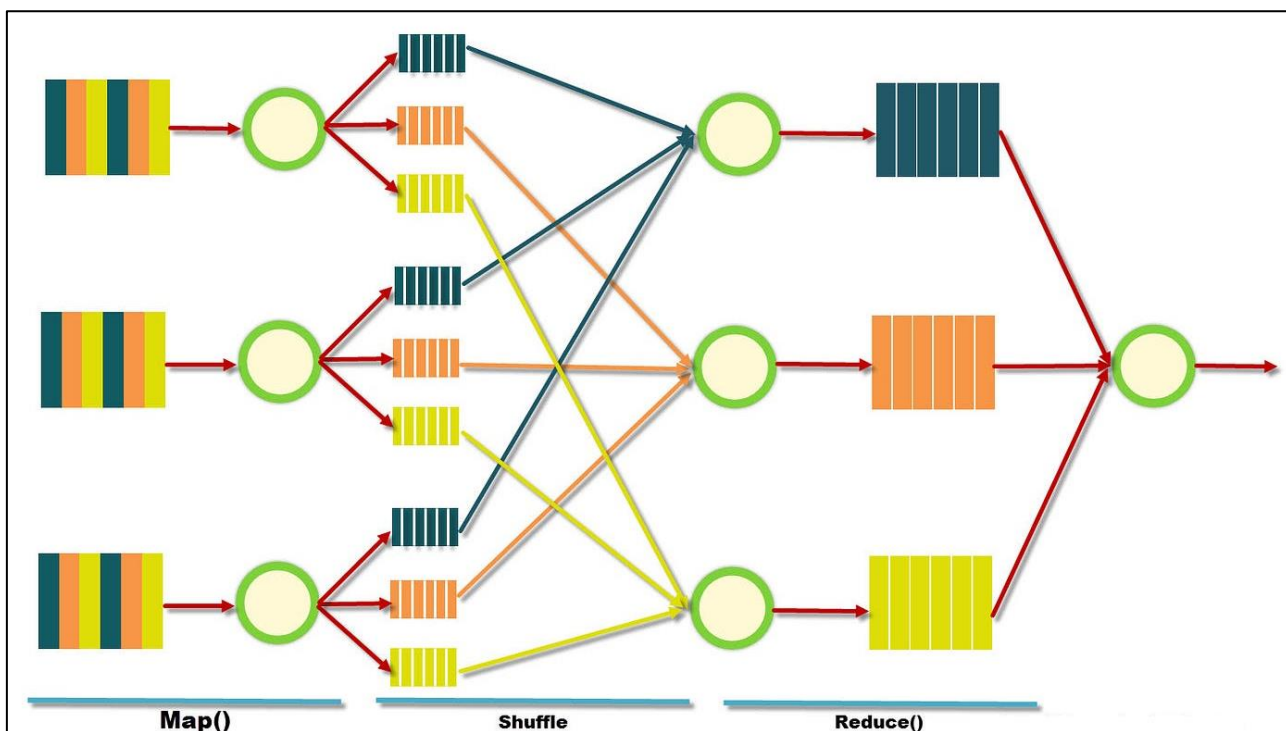


Рисунок 7 – Схема работы MapReduce

Стадия Shuffle проходит незаметно для пользователя, выполняет группировку вывода функции `map()` по ключам. Каждый ключ формирует отдельную "корзину", которая будет входом для функции `reduce()`.

Стадия Reduce принимает каждую "корзину" со значениями, сформированными на стадии Shuffle, и выполняет пользовательскую функцию `reduce()`. Функция `reduce()` вычисляет финальный результат для каждой "корзины". Множество всех значений, возвращаемых функцией `reduce()`, представляет собой окончательный результат задачи MapReduce [67].

MapReduce обладает рядом преимуществ по сравнению с другими технологиями Big Data:

- возможность распределенного выполнения операций предварительной обработки (`map`) и свертки (`reduce`) больших объемов данных, функции `map()` работают независимо и могут выполняться параллельно на разных узлах кластера, аналогично, множество узлов выполняют свертку (`reduce`) после обработки результатов функции `map()` с одним ключом, количество одновременно выполняемых функций `map()` и `reduce()` ограничено доступными ресурсами;

- быстрота обработки больших объемов данных благодаря параллельному выполнению операций, например, MapReduce может отсортировать петабайт данных всего за несколько часов;

- отказоустойчивость и автоматическое восстановление после сбоев, при сбое рабочего узла, выполняющего операции map или reduce, его работа автоматически передается другому доступному узлу с необходимыми входными данными.

MapReduce является эффективным инструментом для обработки больших объемов данных, позволяя распределить вычисления и достичь высокой производительности при работе с Big Data.

#### 2.4.2 Язык программирования Python

Python – высокоуровневый язык программирования общего назначения, который отличается динамической строгой типизацией и автоматическим управлением памятью. Он разработан с учетом улучшения производительности разработчика, читаемости кода, его качества и переносимости программ. Python является полностью объектно-ориентированным языком, где все элементы являются объектами. Одной из необычных особенностей является использование пробельных отступов для выделения блоков кода. Синтаксис ядра Python минималистичен, что позволяет редко обращаться к документации. Python интерпретируемый язык и широко используется для написания скриптов.

Данный язык программирования является одним из ведущих в области аналитики данных и работы с технологией «Большие данные». Так же имеет множество инструментов для работы с ними.

Для работы с языком программирования Python будет использована Visual Studio. Данная среда разработки позволяет разрабатывать различные решения для проектов под управлением Python. На данном языке программирования можно писать, как приложения для компьютеров разных семейств ОС, так и приложения для мобильных платформ и веб-приложения под управлением различных web-framework, таких как: Django, Flask, web2py [53].

### 2.4.3 Выбор базы данных

База данных представляет собой специальное хранилище ценной информации с удобным доступом ко всем данным. Для этого используются системы управления базами данных (СУБД). Одной из популярных СУБД является бесплатная MySQL. Она эффективно работает с веб-приложениями и интернет-сайтами, проста в освоении, что делает ее популярной.

MySQL получила широкую известность благодаря множеству руководств и плагинов, упрощающих работу с ней. Несмотря на отсутствие некоторого функционала других СУБД, MySQL предлагает разнообразные инструменты для создания приложений.

#### Преимущества MySQL:

- простота использования, легкая установка и наличие плагинов для упрощения работы с базами данных;
- обширный функционал, позволяющий реализовать практически любой проект;
- встроенные функции безопасности, обеспечивающие безопасность данных по умолчанию;
- масштабируемость, позволяющая работать с малыми и большими объемами данных;
- высокая производительность благодаря оптимизации используемых стандартов.

#### Недостатки MySQL:

- ограниченная надежность в некоторых процессах работы с данными и недостаточная поддержка функций, таких как связь, транзакции и аудит;
- низкая скорость разработки из-за отсутствия некоторого технического совершенства.

#### Сферы применения MySQL:

- распределенные операции, когда требуется функционал, который отсутствует в других СУБД, например, SQLite;

- обеспечение высокого уровня безопасности;
- работа с интернет-страницами и веб-приложениями, где MySQL является удобной СУБД;
- специфические проекты, где функционал MySQL дает оптимальные результаты.

Однако, есть ситуации, когда MySQL не является подходящим выбором:

- когда требуется полное соответствие стандарту SQL, которое MySQL не обеспечивает полностью;
- проект предусматривает многопоточность данных, что может вызывать проблемы при параллельных операциях чтения/записи;
- имеющийся функционал MySQL не удовлетворяет всем требованиям работы с базой данных.

## 3 ОСНОВНЫЕ ЭТАПЫ ПРАКТИЧЕСКОЙ РАЗРАБОТКИ ПРОГРАММНОГО ПРОДУКТА

Разработку программного продукта можно разбить на несколько этапов такие как, создание архитектурного проекта, проектирование базы данных, разработка алгоритма для сбора данных, алгоритмов для создания моделей машинного обучения, выбор архитектуры программного продукта, создание пользовательского интерфейса, создание серверной части приложения. Далее будут подробно описаны перечисленные выше этапы в подробном формате.

### 3.1 Архитектурный проект

В данной пункте описана детальная разработка программного продукта. Один из ключевых аспектов при проектировании программного решения – это понимание требований пользователей и функциональных возможностей, которые необходимо предоставить. Для достижения этой цели использован набор инструментов и методов, включающих в себя диаграмму вариантов использования, диаграмму последовательности, диаграмму деятельности и функциональную модель.

Все эти диаграммы являются мощными инструментами для проектирования программных решений. Они позволяют увидеть полную картину работы программы, взаимодействие с пользователями и внутреннюю структуру системы. Их создание и анализ позволяют выявить потенциальные проблемы, оптимизировать процессы и обеспечить качественное функционирование программного продукта.

#### 3.1.1 Диаграмма прецедентов

Для того чтобы полноценно разработать и спроектировать программный продукт, необходимо тщательно изучить потребности и ожидания пользователей. Одним из инструментов, который помогает в этом процессе, является диаграмма вариантов использования.



Диаграмма вариантов использования представляет собой графическое описание функциональных требований к программному продукту. Она позволяет визуализировать взаимодействие между актерами (пользователями) и системой, а также описывает различные сценарии использования продукта.

Прецедент в диаграмме вариантов использования представляет собой определенное действие или операцию, которую пользователь выполняет в системе. Он описывает конкретную функциональность, которую система предоставляет пользователю для достижения определенной цели.

Каждый прецедент описывает взаимодействие между актерами и системой в рамках определенного сценария использования. Прецеденты могут быть различными по своей сложности и уровню детализации. Они помогают понять, как пользователи будут взаимодействовать с системой и какие функциональности им необходимы для выполнения своих задач.

В программе информационной поддержки киберспортивных команд в качестве актера выступает пользователь. В качестве прецедентов выделены следующие функциональные части системы:

- получение информации о командах;
- получение информации об игроках;
- получение информации о героях;
- предложение контрпика;
- предложение дополнения пика;
- запрос в базу данных;
- использование модели машинного обучения;
- предоставление данных пользователю.

Диаграмма вариантов использования программы представлена на рисунке 8.

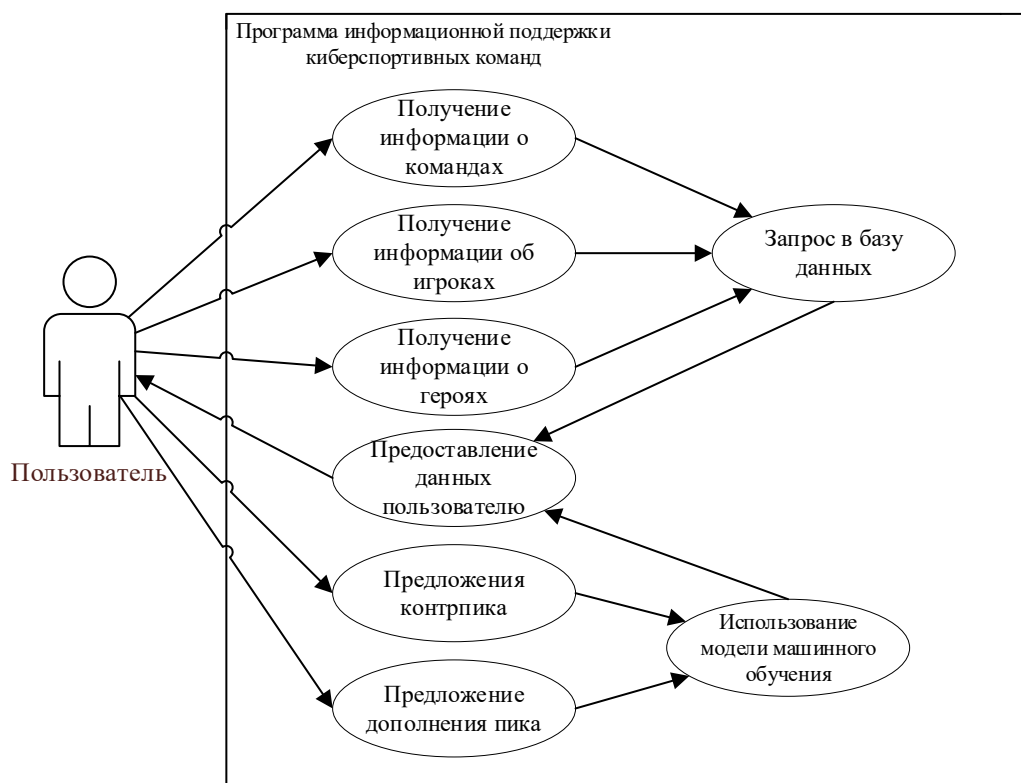


Рисунок 8 – Диаграмма вариантов использования

Описание основных прецедентов:

- получение информации о командах:
  - входные данные: запрос пользователя о команде;
  - основные шаги: извлечение информации о командах из базы данных, предоставление пользователю запрошенной информации;
- получение информации об игроках:
  - входные данные: запрос пользователя об игроке;
  - основные шаги: извлечение информации об игроке из базы данных, предоставление пользователю запрошенной информации;
- получение информации о герое:
  - входные данные: запрос пользователя о герое;
  - основные шаги: извлечение информации о герое из базы данных, предоставление пользователю запрошенной информации;
- предложение контрпики:
  - входные данные: выбранный пользователем герой;

- основные шаги: использование модели машинного обучения для предсказания контрпика, предоставление пользователю рекомендованного контрпика;
- предложение дополнения пика:
  - входные данные: 2-4 выбранных пользователем героев;
  - основные шаги: использование модели машинного обучения для предсказания дополнения пика, предоставление пользователю рекомендованного дополнения пика;

В данной пункте была представлена диаграмма вариантов использования, которая является мощным инструментом для описания функциональных требований к программному продукту. В процессе анализа и проектирования программного продукта были выделены прецеденты, описывающие конкретные сценарии использования системы.

### 3.1.2 Диаграмма последовательности

Диаграмма последовательности является графическим представлением взаимодействия различных объектов или актеров в системе в определенной последовательности действий. Она помогает визуализировать поток сообщений или вызовов между объектами и понять, как происходит выполнение определенных сценариев.

Для программы, которая собирает данные о киберспортивных матчах и предлагает контрпики и дополнения пикам на основе модели машинного обучения, диаграмма последовательности может быть полезным инструментом для иллюстрации взаимодействия различных компонентов системы во время выполнения определенных сценариев.

Диаграмма последовательности включает следующие элементы:

- актеры: актеры представляют различные роли или пользователей, которые взаимодействуют с системой, на диаграмме актером является пользователь, с которым программа взаимодействует;
- объекты: объекты представляют компоненты или модули системы, которые выполняют определенные функции, на диаграмме объектами будут

программный продукт, модель машинного обучения, база данных, парсер и Web-ресурс;

– сообщения: представляют взаимодействие между актерами и объектами, они отображают передачу данных, вызовы методов или выполнение операций между различными компонентами системы.

Диаграмма последовательности представлена на рисунке 9.

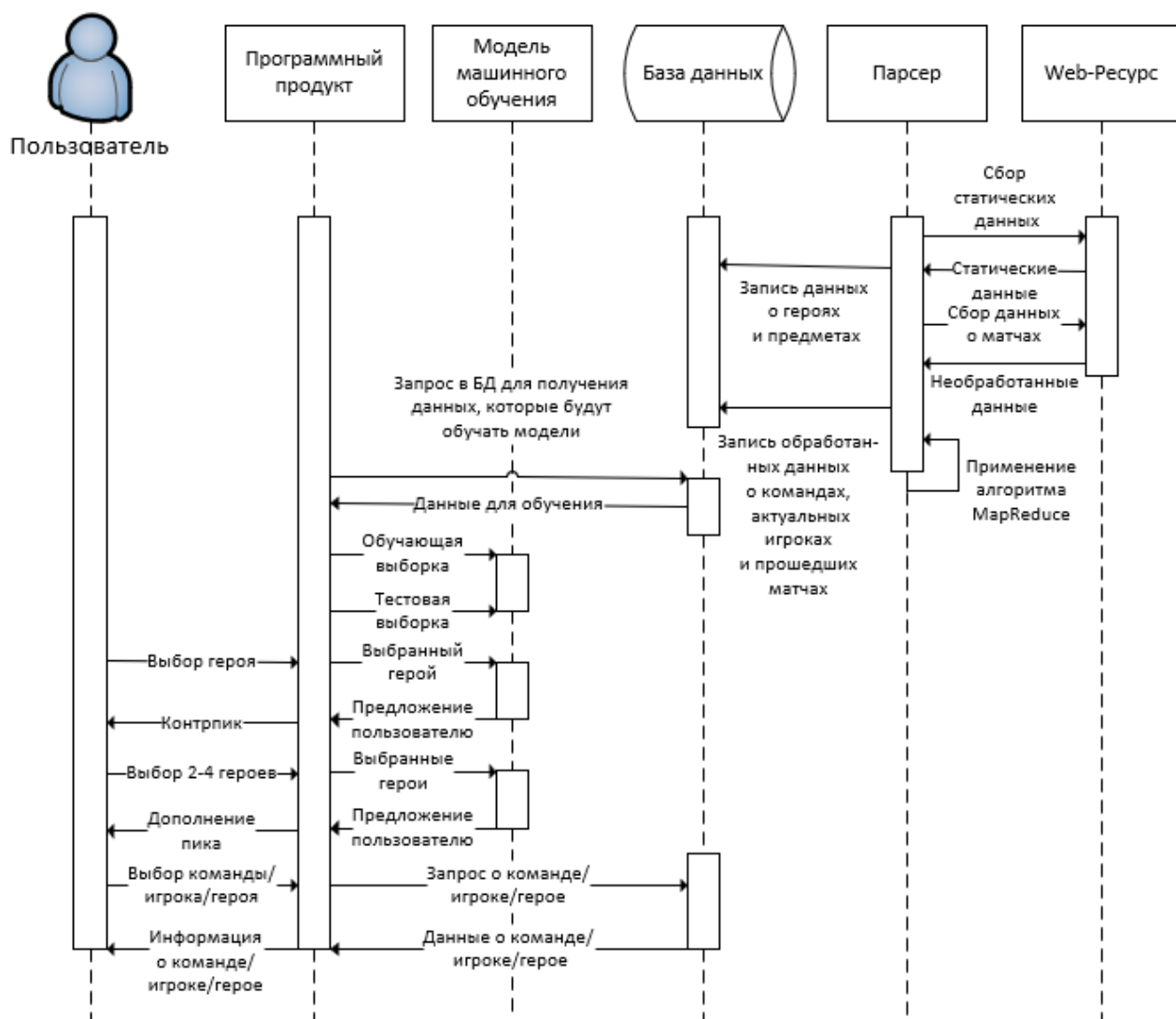


Рисунок 9 – Диаграмма последовательности

Диаграмма последовательности использована для описания различных сценариев использования программы, например, процесса сбора данных, обучения модели машинного обучения, предложения контрпика, дополнения пика и других функциональностей системы.

Путем анализа и создания диаграммы последовательности возможно лучше понять взаимодействие компонентов системы, последовательность выполнения операций и поток данных между ними. Это поможет оптимизировать процессы, выявить возможные проблемы и улучшить работу программного продукта в целом.

### 3.1.3 Диаграмма деятельности

Диаграмма деятельности представляет собой графическое представление последовательности действий или процессов, которые выполняются в системе или в рамках определенной функциональности. Она помогает визуализировать поток работы или операций, а также условия и ветвления в процессе выполнения.

Диаграмма деятельности программы при взаимодействии с внешними ресурсами на рисунке 10.

Диаграмма деятельности программы при взаимодействии с внешними ресурсами содержит 2 основные точки системы: точка старта (начало) и точка окончания (конец). Точка старта – начало работы алгоритма программы, точка окончания, соответственно – конец работы алгоритма.

После старта работы программы производится проверка подключения к серверу (блок «Проверка подключение к серверу»). При отсутствии подключения программа переходит в состояние ошибки (блок «Сообщение об ошибке»). Если подключение к серверу присутствует, то переход к началу работы (блок «Начало работы пользователя»).

Следующий шаг – выбор действия. Программа находится в состоянии ожидания действия пользователя. После выбора действия пользователем происходит переход к определению системой внешнего ресурса, с которым будет происходить взаимодействие. Если действие связано с запросом к базе данных, то переход к формированию запроса (блок «Формирование запроса в БД»).

Если выбор пользователя связан с обращением к обученной модели, то переход к предоставлению данных модели (блок «Представление данных обученной модели»).

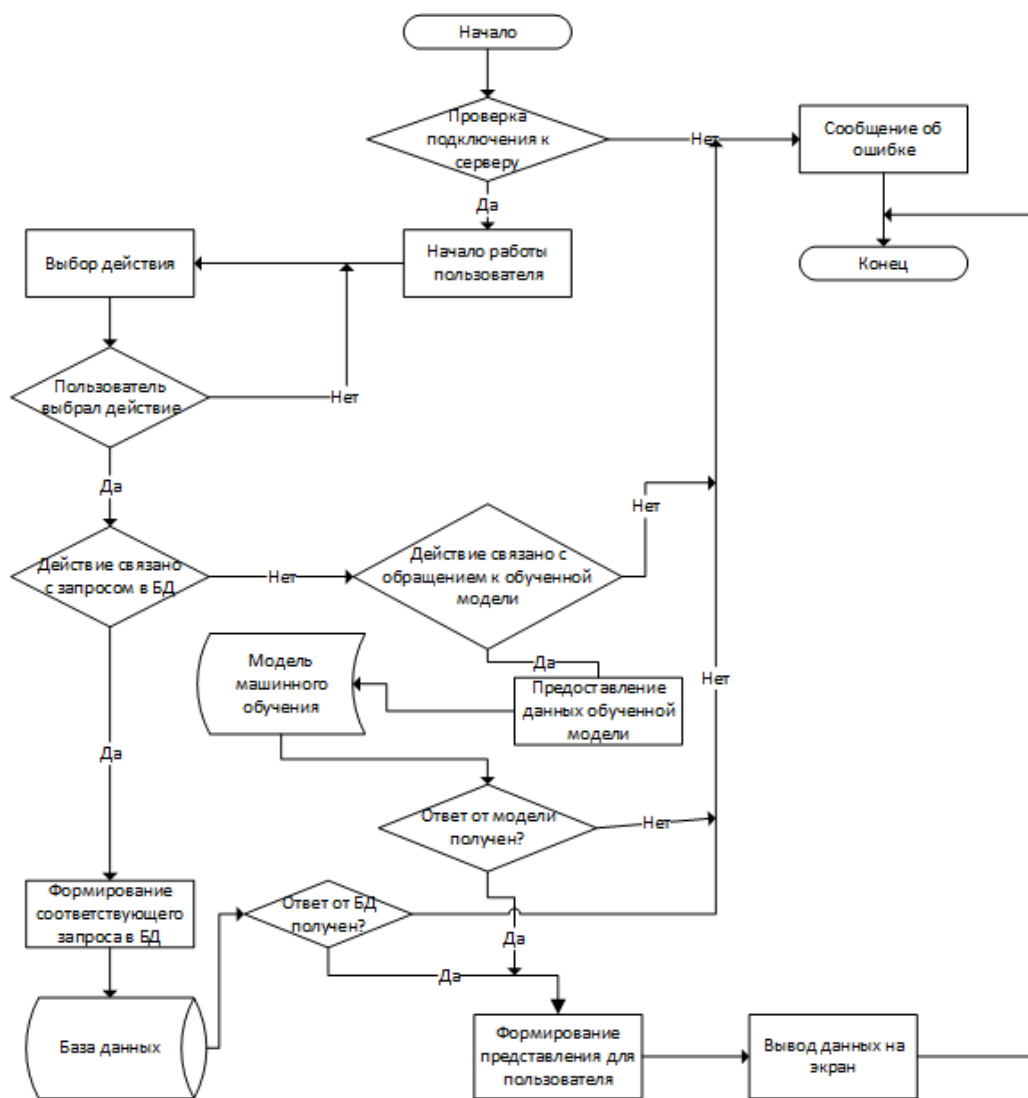


Рисунок 10 – Диаграмма деятельности программы при взаимодействии с внешними ресурсами

Если выбор пользователя не связан с обращением к базе данных или модели машинного обучения, то программа выводит сообщение об ошибке.

После в зависимости от выбора пользователя алгоритм программы может пойти по двум путям, которые после отработки вернуться к общему знаменателю.

Если ответ от базы данных или обученной модели не получен, то программа выводит сообщение об ошибке.

Если ответ от базы данных или обученной модели получен, то программа переходит к этапу формирования представления для пользователя (блок «Формирование представления для пользователя»). Сформированные данные

выводятся на экран пользователя. Программа переходит к точке окончания работы (блок «Конец»).

Создание диаграммы деятельности помогает визуализировать и лучше понять последовательность операций, условия и ветвления в процессе выполнения.

#### 3.1.4 Функциональная модель

Функциональная модель является важной частью архитектурного проекта, поскольку она позволяет визуализировать основные компоненты и их взаимодействие в рамках программы.

Функциональная модель опирается на концепцию функциональности программы и выделяет ключевые компоненты, входные и выходные данные, а также механизмы и управление программой. Она помогает понять, как различные части программы работают вместе, какие данные они обрабатывают и как они взаимодействуют с внешними системами.

Для начала необходимо выделить входные и выходные данные, а также элементы управления и механизмы, на функциональной модели они выделяются как стрелки к блоку программы. Стрелки слева – входные данные, справа – выходные, сверху – управление, снизу – механизмы.

Входными данными были выделены:

- команды пользователя;
- данные о командах, игроках и героях;
- данные о прошедших киберспортивных соревнованиях.

Механизмы и компоненты:

- web-ресурс: этот компонент представляет собой внешний ресурс, с которого собираются данные о прошедших киберспортивных матчах, он содержит информацию о командах, игроках, героях, результатах матчей и других сущностях, необходимых для функционирования программы;
- парсер: этот компонент отвечает за сбор и преобразование данных, полученных с web-ресурса, он выполняет анализ HTML-страниц или других источников данных, извлекает необходимую информацию и преобразует ее в удобный формат для обработки программой;

- база данных: этот компонент представляет собой хранилище, где сохраняются собранные и обработанные данные, она используется для долгосрочного хранения информации о матчах, командах, игроках и других сущностях, обеспечивает доступ к данным и возможность выполнения запросов для извлечения необходимой информации;

- модель машинного обучения: этот компонент отвечает за обучение модели на основе собранных данных, модель используется для предложения контрпика и дополнения пика на основе анализа и предсказания результатов матчей, принимает входные данные, выполняет обучение на основе этих данных и генерирует соответствующие рекомендации;

- интерфейс программы: этот компонент представляет собой пользовательский интерфейс, через который пользователь может взаимодействовать с программой, позволяет пользователю получать информацию о командах, игроках, героях, а также использовать модули предложения контрпика и дополнения пика, основанные на модели машинного обучения.

#### Управление:

- алгоритм MapReduce: выполняет ключевую роль в обработке данных и обеспечении эффективного параллельного вычисления, отвечает за отделения неактуальных данных, этот механизм обеспечивает быструю и масштабируемую обработку данных, что является важным элементом функциональной модели;

- пользовательский ввод: представляет собой взаимодействие пользователей с программным продуктом через интерфейс, пользователи могут вводить различные команды и запросы, такие как выбор героя, получение информации о командах и игроках, а также управление процессом предложения контрпика и дополнения пика, является важным механизмом управления, который определяет действия и потребности пользователей и направляет работу программы в соответствии с этими потребностями;

- градиентный бустинг: является механизмом машинного обучения, который используется для обучения модели и предложения контрпика и дополнения пика, обеспечивает анализ и предсказание на основе имеющихся данных



о матчах и героях, играет важную роль в процессе принятия решений, предоставляя программе способность предлагать оптимальные варианты героев, соответствующих требованиям пользователей.

Выходные данные:

– рекомендации контрпика: этот результат представляет собой список рекомендованных героев, которые могут быть эффективными контрпиками выбранному пользователем герою, они основаны на анализе данных о прошедших матчах и модели машинного обучения, этот вывод помогает пользователю принять более осознанное решение при выборе героя для битвы.

– рекомендации дополнения пика: этот результат представляет собой список рекомендованных героев, которые могут быть эффективным дополнением к уже выбранным героям пользователем, рекомендации дополнения пика также основаны на анализе данных о матчах и модели машинного обучения, они помогают пользователю формировать более сбалансированные и сильные команды для игры.

– информация о командах, игроках и героях: этот вывод предоставляет пользователю доступ к различной информации, связанной с командами, игроками и героями, включает данные о статистике команд, их истории побед и поражений, профилях игроков, а также характеристики героев, их уникальные способности.

Функциональная модель программы представлена на рисунке 11.

Описание функциональной модели программы предоставляет общее представление о ее функциональности, входных и выходных данных, механизмах и управлении. Это позволяет понять, как различные компоненты взаимодействуют между собой и как пользователь будет взаимодействовать с программой.

Теперь необходимо перейти к следующему шагу - декомпозиции функциональной модели.

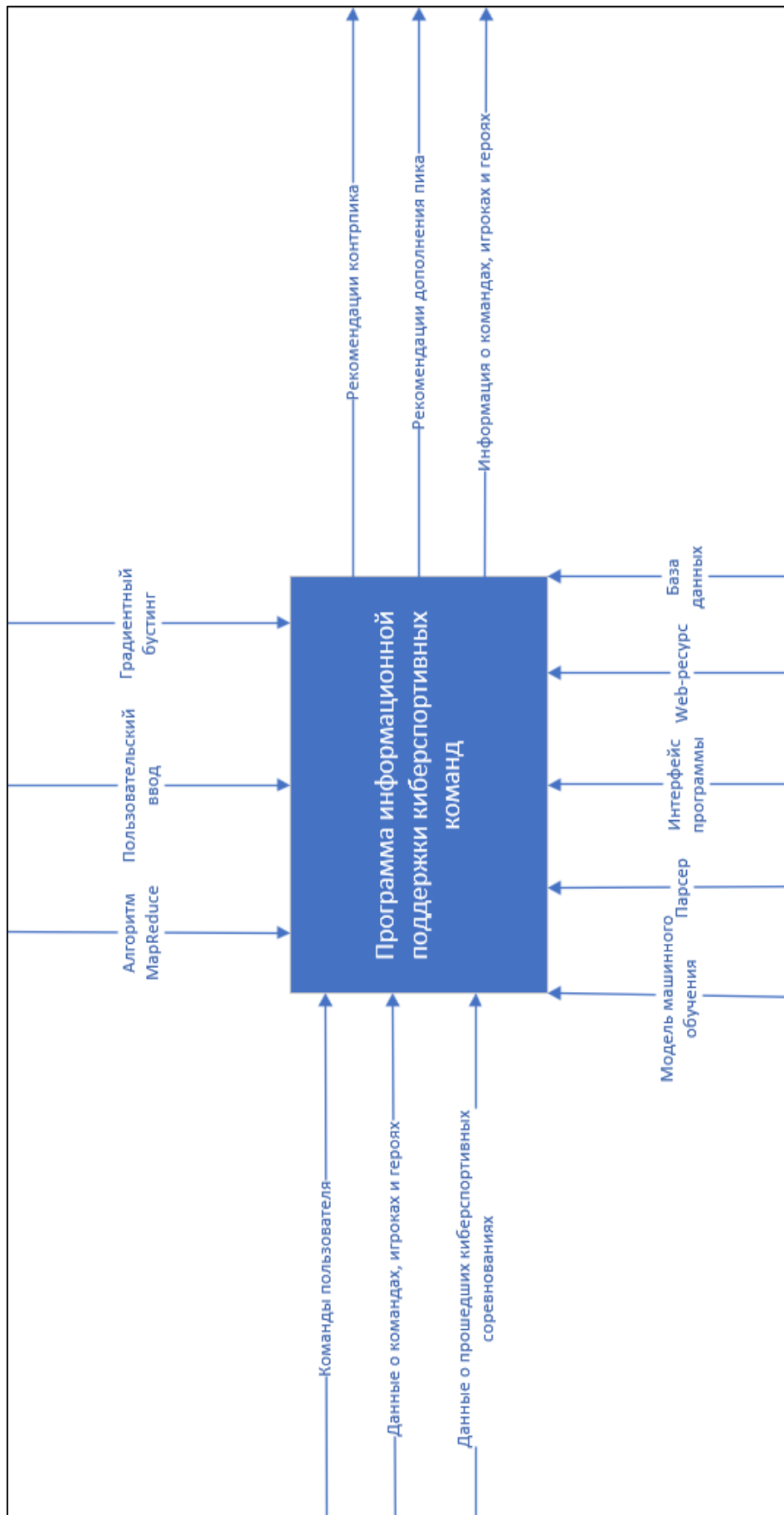


Рисунок 11 – Функциональная модель программы

Декомпозиция функциональной модели программного продукта представляет собой разбиение его функциональности на отдельные модули, каждый из которых выполняет определенные задачи.

В рамках данной декомпозиции выделены следующие модули:

- модуль сбора данных: здесь реализован парсер, который осуществляет сбор данных о прошедших киберспортивных матчах с web-ресурса, данные, полученные парсером, передаются на следующий модуль для дальнейшей обработки;
- модуль обработки данных: тут применяется алгоритм MapReduce для обработки и фильтрации собранных данных, модуль позволяет избавиться от неактуальных данных и подготовить их для последующего использования;
- модуль сохранения данных в БД: данный модуль сохраняет данные в базу данных, обеспечивает надежное хранение и управление этими данными, что позволяет эффективно работать с ними в дальнейшем.
- модуль подготовки данных к обучению: выполняет предварительную обработку данных, необходимую для обучения модели машинного обучения, то есть разделяет их на выборки;
- модуль обучения модели: этот модуль обучает модель на подготовленных данных, обученная модель будет использоваться для предложения рекомендаций контрпика и дополнения пика;
- модуль пользовательского интерфейса: здесь реализуется интерфейс программы, который позволяет пользователям взаимодействовать с программой;
- модуль рекомендаций: данный модуль реализует функции предложения рекомендаций, модель машинного обучения используется для предложения контрпика выбранному пользователем герою и дополнения пика на основе выбранных пользователем героев.

Декомпозиция функциональной модели представлена на рисунке 12.

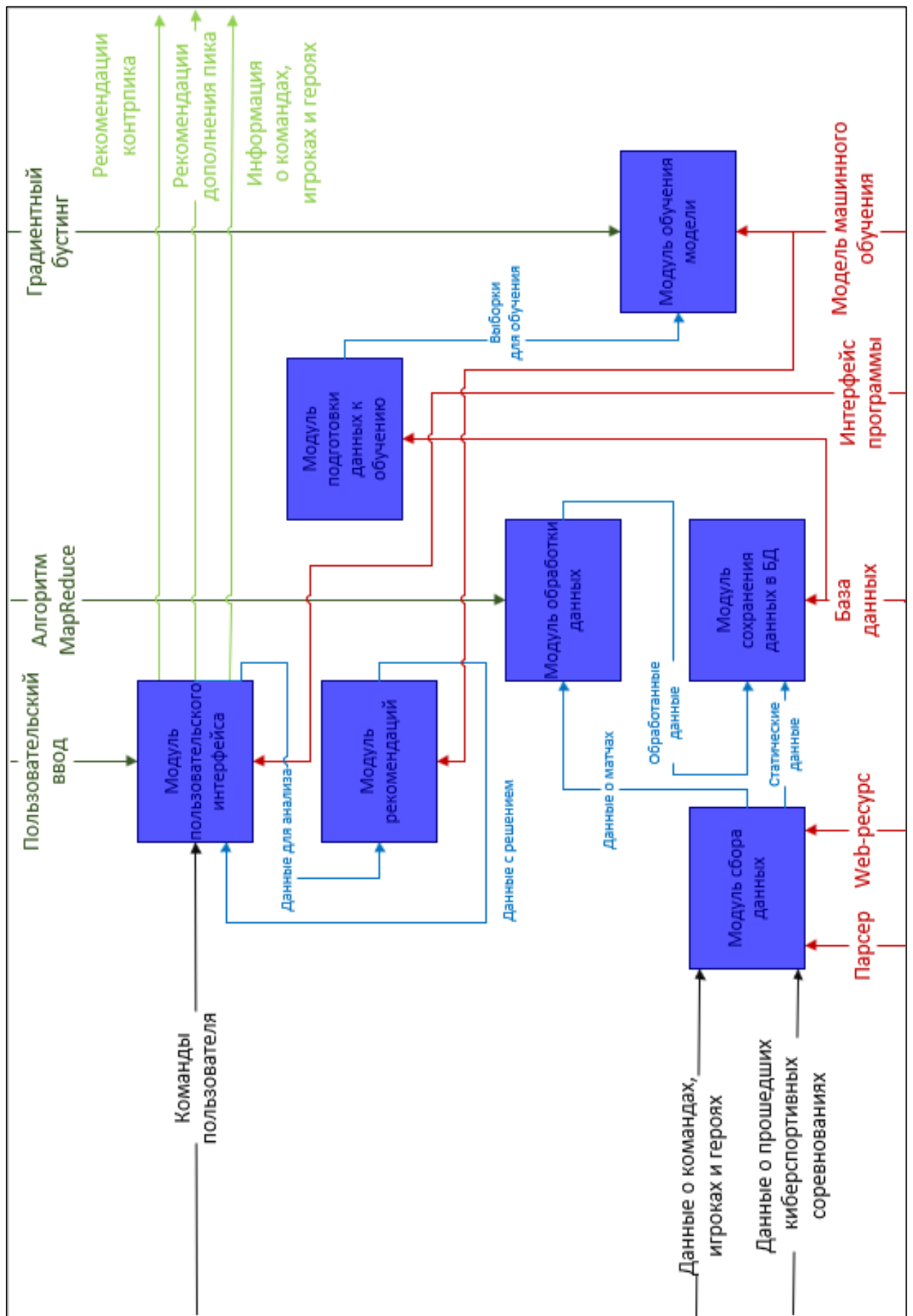


Рисунок 12 – Декомпозиция функциональной модели программы

Декомпозиция модели позволяет разбить сложную систему на более мелкие и управляемые компоненты, что способствует легкости в разработке, тестировании и поддержке программного продукта. Каждый модуль выполняет определенные задачи, взаимодействует с другими модулями и обеспечивает определенный функционал системы.

Такой подход к декомпозиции модели позволяет более четко определить ответственность каждого модуля, улучшить переиспользуемость кода, облегчить сопровождение и расширение системы в будущем. Кроме того, это позволяет более эффективно организовать работу разработчиков, распределить задачи и обеспечить согласованность работы.

### **3.2 Проектирование базы данных**

Для последующих этапов разработки необходимо точно определить структуру базы данных. Так как большинство алгоритмов реализуемых в программном продукте будут работать с структурой базы данных.

Без правильно спроектированной базы данных алгоритмы сбора данных и алгоритмы предложения решений будут работать некорректно.

#### **3.2.1 Инфологическое проектирование**

При разработке базы данных для программы информационной поддержки киберспортивных команд были выделены следующие сущности:

- сущность «Команды» содержит информацию о всех киберспортивных командах, участвующих на профессиональных турнирах в дисциплине «DOTA 2»;
- сущность «Игроки» содержит информацию о всех игроках, которые участвовали в матчах профессиональных турниров;
- сущность «Герои» содержит информацию о списке героев, которые доступны для выбора профессиональных команд в игре;
- сущность «Предметы» содержит информацию об игровых предметах, которые доступны для покупки в игре;
- сущность «Турниры» содержит информацию о профессиональных турнирах, проводимых в киберспортивной дисциплине «DOTA 2»;

- сущность «Турниры\_команды» содержит информацию о списке команд, участвующих на определенных профессиональных турнирах;
- сущность «Матчи» содержит информацию о результатах киберспортивных матчей;
- сущность «Матчи\_герои» содержит информацию о героях, выбранных в киберспортивных матчах;
- сущность «Матчи\_предметы» содержит информацию о купленных предметах для героев в киберспортивных матчах.

В таблицах 2-10 представлены спецификации атрибутов для каждой сущности.

Таблица 2 – Спецификация атрибутов сущности «Команды»

Название атрибута	Описание атрибута	Тип данных	Диапазон значений	Пример атрибута
<u>Код_команды</u>	Число, однозначно определяющее каждую команду	Числовой	>0	1
Наименование	Название команды	Текстовой	-	Alliance
Год_образования	Точная дата образования киберспортивной организации	Дата	-	14.12.2012
Менеджер	Игровой никнейм менеджера организации	Текстовой	-	Loda

Таблица 3 – Спецификация атрибутов сущности «Игроки»

Название атрибута	Описание атрибута	Тип данных	Диапазон значений	Пример атрибута
1	2	3	4	5
<u>Код_игрока</u>	Число, однозначно определяющее каждого игрока	Числовой	>0	1

1	2	3	4	5
Никнейм	Никнейм игрока	Текстовой	-	s4
Страна	Гражданство игрока	Текстовый	-	Швеция
Роль	Роль игрока в игре	Текстовой	-	Offlane

Таблица 4 – Спецификация атрибутов сущности «Герои»

Название атрибута	Описание атрибута	Тип данных	Диапазон значений	Пример атрибута
<u>Код_героя</u>	Число, однозначно определяющее каждого героя	Числовой	>0	1
Имя героя	Внутриигровое название героя	Текстовый	-	Ogre mage
Способность1	Наименование первой способности героя	Текстовой	-	Fireblast
Способность2	Наименование второй способности героя	Текстовый	-	Ignite
Способность3	Наименование третьей способности героя	Текстовой	-	Bloodlust
Способность4	Наименование четвертой способности героя	Текстовый	-	Multicast

Таблица 5 – Спецификация атрибутов сущности «Предметы»

Название атрибута	Описание атрибута	Тип данных	Диапазон значений	Пример атрибута
1	2	3	4	5
<u>Код_предмета</u>	Число, однозначно определяющее каждый предмет	Числовой	>0	1

1	2	3	4	5
Название предмета	Внутриигровое название предмета	Текстовый	-	Power Treads
Стоимость	Стоимость предмета для покупки	Числовой	-	1400
Описание предмета	Внутриигровое описание предмета	Текстовый	-	Пара сапог, выделанных из жёсткой кожи, способна изменять свои свойства по желанию владельца.

Таблица 6 – Спецификация атрибутов сущности «Турниры»

Название атрибута	Описание атрибута	Тип данных	Диапазон значений	Пример атрибута
<u>Код_турнира</u>	Число, однозначно определяющее каждый турнир	Числовой	>0	1
Наименование турнира	Официальное название турнира	Текстовый	-	ESL One Stockholm Major 2022
Формат турнира	Формат проведения турнира	Текстовой	-	LAN
Призовой фонд	Призовой фонд турнира	Текстовый	-	\$500,000
Количество команд	Количество команд, участвующих на турнире	Числовой	-	14
Место проведения	Название города, в котором проводится финальная часть турнира	Текстовый	-	Stockholm



Таблица 7 – Спецификация атрибутов сущности «Турниры\_команды»

Название атрибута	Описание атрибута	Тип данных	Диапазон значений	Пример атрибута
<u>Код_турниры_команды</u>	Число, однозначно определяющее каждую запись в таблице	Числовой	>0	1
Место команды	Место в финальной таблице турнира определённой команды	Числовой	-	16

Таблица 8 – Спецификация атрибутов сущности «Матчи»

Название атрибута	Описание атрибута	Тип данных	Диапазон значений	Пример атрибута
<u>Код_матча</u>	Число, однозначно определяющее каждый матч	Числовой	>0	1
Результат	Результат матча	Текстовый	-	Победа сил света
Убийства сил света	Количество совершенных убийств силами света	Числовой	-	62
Убийства сил тьмы	Количество совершенных убийств силами тьмы	Числовой	-	61
Длительность	Продолжительность матча	Время	-	59:30

Таблица 9 – Спецификация атрибутов сущности «Матч\_герои»

Название атрибута	Описание атрибута	Тип данных	Диапазон значений	Пример атрибута
1	2	3	4	5

Продолжение таблицы 9

1	2	3	4	5
Код_матч_герои _____	Число, одно-значно определяющее героя в матче	Числовой	>0	1
Убийства	Количество убийств	Числовой	-	23
Помощи	Количество помощи	Числовой	-	12
Смерти	Количество смертей	Числовой	-	2
ГПМ	Средний показатель заработанного золота в минуту	Числовой	-	864
ХПМ	Средний показатель заработанного опыта в минут	Числовой	-	745
Урон по героям	Количество нанесенного урона по вражеским героям	Числовой	-	54953
Урон по строениям	Количество нанесенного урона по вражеским строениям	Числовой	-	10423
Добито крипов	Количество добытый нейтральных крипов	Числовой	-	544
Лечение	Количество выданного лечения союзных героев	Числовой	-	0

Таблица 10 – Спецификация атрибутов сущности «Матч\_предметы»

Название атрибута	Описание атрибута	Тип данных	Диапазон значений	Пример атрибута
Код_матч_предметы	Число, однозначно определяющее каждый предмет в матче	Числовой	>0	1
Время покупки	Время покупки внутриигрового предмета	Время	-	15:12

Связи между сущностями отображают в информационной модели то, как взаимосвязаны между собой объекты предметной области. Все взаимосвязи представлены в таблице 11.

Таблица 11 – Связи между сущностями

Название первой сущности, участвующей в связи	Название второй сущности, участвующей в связи	Название связи	Тип связи	Обоснование выбора типа связи
1	2	3	4	5
Команды	Игроки	Соответствует	Один ко многим	Каждой записи в таблице «Игроки» соответствует определенная запись из таблицы «Команды».
Турниры	Турниры_команды	Соответствует	Один ко многим	Каждой записи в таблице «Турниры_команды» соответствует определенная запись из таблицы «Турниры».
Команды	Турниры_команды	Соответствует	Один ко многим	Каждой записи в таблице «Турниры_команды» соответствует определенная запись из таблицы «Команды».

1	2	3	4	5
Команды	Матчи	Соответствует	Многие ко многим	Каждой записи в таблице «Матчи» соответствуют 2 записи из таблицы «Команды»
Турниры	Матчи	Соответствует	Один ко многим	Каждой записи в таблице «Матчи» соответствует определенная запись из таблицы «Турниры».
Матчи	Матч_герои	Соответствует	Один ко многим	Каждой записи в таблице «Матч_герои» соответствует определенная запись из таблицы «Матчи».
Герои	Матч_герои	Соответствует	Один ко многим	Каждой записи в таблице «Матч_герои» соответствует определенная запись из таблицы «Герои».
Игроки	Матч_герои	Соответствует	Один ко многим	Каждой записи в таблице «Матч_герои» соответствует определенная запись из таблицы «Игроки».
Матчи	Матч_предметы	Соответствует	Один ко многим	Каждой записи в таблице «Матч_предметы» соответствует определенная запись из таблицы «Матчи».
Герои	Матч_предметы	Соответствует	Один ко многим	Каждой записи в таблице «Матч_предметы» соответствует определенная запись из таблицы «Герои».
Игроки	Матч_предметы	Соответствует	Один ко многим	Каждой записи в таблице «Матч_предметы» соответствует определенная запись из таблицы «Игроки».

1	2	3	4	5
Предметы	Матч_пред- меты	Соответ- ствует	Один ко мно- гим	Каждой записи в таб- лице «Матч_пред- меты» соответствует определенная запись из таблицы «Пред- меты».

### 3.2.2 Логическое проектирование

Отображение концептуально инфологической модели является важным этапом в проектировании БД. Отображения выполняются для каждой пары сущностей.

Начнем с сущностей «Команды» и «Игроки». Связь между ними – один ко многим.



Рисунок 13 – Связь «Команды – Игроки»

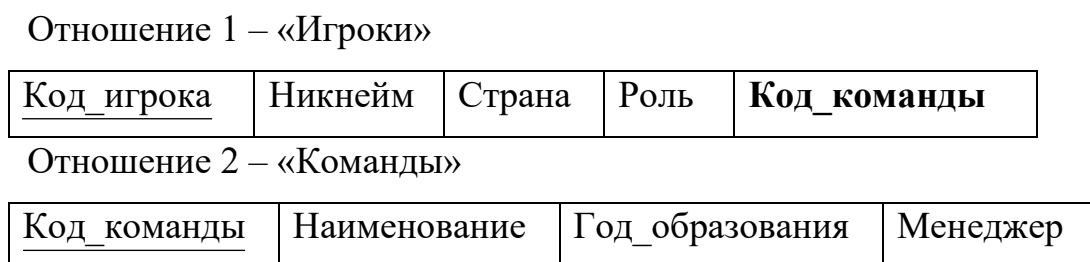


Рисунок 14 – Отношения 1 и 2

Следующие сущности – «Турниры» и «Турниры\_команды». Связь между ними – один ко многим.

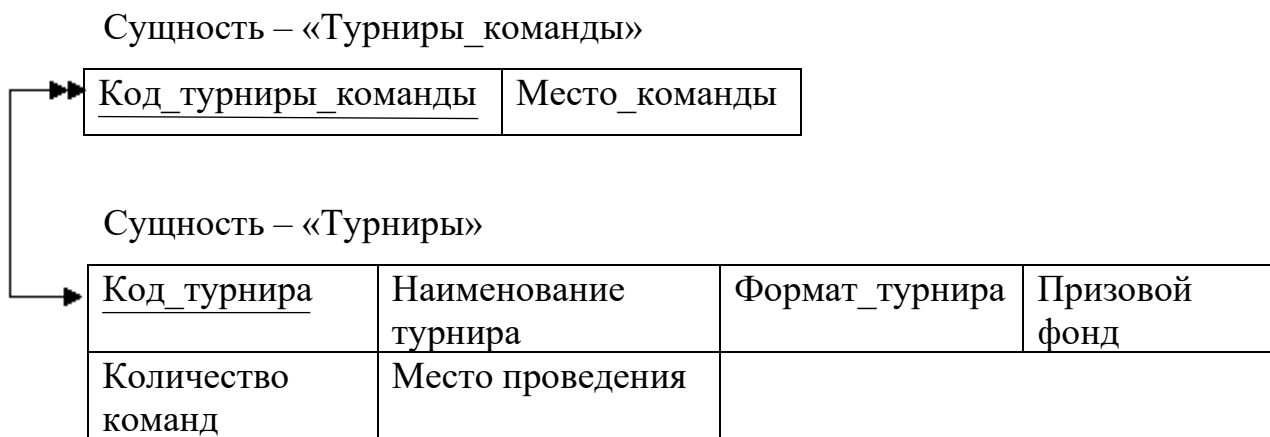


Рисунок 15 – Связь «Турниры\_команды – Турниры»



Рисунок 16 – Отношения 3 и 4

Следующие сущности – «Команды» и «Турниры\_команды». Связь между ними – один ко многим.



Рисунок 17 – Связь «Команды – Турниры\_команды»

Отношение 5 – «Турниры\_команды»

<u>Код_турниры_команды</u>	Место_команды	Код_турнира	<b>Код_команды</b>
----------------------------	---------------	-------------	--------------------

Отношение 6 – «Команды»

<u>Код_команды</u>	Наименование	Год_образования	Менеджер
--------------------	--------------	-----------------	----------

Рисунок 18 – Отношения 5 и 6

Следующие сущности – «Команды» и «Матчи». Связь между ними – многие ко многим.

Сущность – «Матчи»

<u>Код_матча</u>	Результат	Убийств сил света	Убийств сил тьмы	Длительность
------------------	-----------	----------------------	---------------------	--------------

Сущность – «Команды»

<u>Код_команды</u>	Наименование	Год_образования	Менеджер
--------------------	--------------	-----------------	----------

Рисунок 19 – Связь «Команды – Матчи»

Отношение 7 – «Матчи»

<u>Код_матча</u>	Результат	Убийств сил света	Убийств сил тьмы	Длительность
<b>Код_команды1</b>	<b>Код_команды2</b>			

Отношение 8 – «Команды»

<u>Код_команды</u>	Наименование	Год_образования	Менеджер
--------------------	--------------	-----------------	----------

Рисунок 20 – Отношения 7 и 8

Следующие сущности – «Турниры» и «Матчи». Связь между ними – один ко многим.

Сущность – «Матчи»

<u>Код_матча</u>	Результат	Убийств сил света	Убийств сил тьмы	Длитель- ность
Код_команды1	Код_команды2			

Сущность – «Турниры»

<u>Код_турнира</u>	Наименование турнира	Формат_турнира	Призовой фонд
Количество команд	Место проведения		

Рисунок 21 – Связь «Матчи – Турниры»

Отношение 9 – «Матчи»

<u>Код_матча</u>	Результат	Убийств сил света	Убийств сил тьмы	Длитель- ность
Код_команды1	Код_команды2	<b>Код_турнира</b>		

Отношение 10 – «Турниры»

<u>Код_турнира</u>	Наименование турнира	Формат_турнира	Призовой фонд
Количество команд	Место проведения		

Рисунок 22 – Отношения 9 и 10

Следующие сущности – «Матч» и «Матч\_герои». Связь между ними – один ко многим.

Сущность – «Матч\_герои»

<u>Код_матча_героя</u>	Убийства	Помощи	Смерти	ГПМ	ХПМ
Урон по героям	Урон по строениям	Добито крипов	Лечение		

Сущность – «Матчи»

<u>Код_матча</u>	Результат	Убийств сил света	Убийств сил тьмы	Длитель- ность
Код_команды1	Код_команды2	Код_турнира		

Рисунок 23 – Связь «Матч\_герои – Матчи»



Отношение 11 – «Матч\_герои»

<u>Код_матча_героя</u>	Убийства	Помощи	Смерти	ГПМ	ХПМ
Урон по героям	Урон по строениям	Добито крипов	Лечение	<b>Код_матча</b>	

Отношение 12 – «Матчи»

<u>Код_матча</u>	Результат	Убийств сил света	Убийств сил тьмы	Длительность
Код_команды1	Код_команды2	Код_турнира		

Рисунок 24 – Отношения 11 и 12

Следующие сущности – «Герои» и «Матч\_герои». Связь между ними – один ко многим.

Сущность – «Матч\_герои»

<u>Код_матча_героя</u>	Убийства	Помощи	Смерти	ГПМ	ХПМ
Урон по героям	Урон по строениям	Добито крипов	Лечение	Код_матча	



Сущность – «Герои»

<u>Код_героя</u>	Имя_героя	Способность1	Способность2
Способность3	Способность4		

Рисунок 25 – Связь «Герои – Матч\_герои»

Отношение 13 – «Матч\_герои»

<u>Код_матча_героя</u>	Убийства	Помощи	Смерти	ГПМ	ХПМ
Урон по героям	Урон по строениям	Добито крипов	Лечение	Код_матча	
<b>Код_героя</b>					

Отношение 14 – «Герои»

<u>Код_героя</u>	Имя_героя	Способность1	Способность2
Способность3	Способность4		

Рисунок 26 – Отношения 13 и 14

Следующие сущности – «Игроки» и «Матч\_герои». Связь между ними – один ко многим.

Сущность – «Матч\_герои»

<u>Код_матча_героя</u>	Убийства	Помощи	Смерти	ГПМ	ХПМ
Урон по героям	Урон по строениям	Добито крипов	Лечение	Код_матча	
Код_героя					

Сущность – «Игроки»

<u>Код_игрока</u>	Никнейм	Страна	Роль	Код_команды
-------------------	---------	--------	------	-------------

Рисунок 27 – Связь «Игроки – Матч\_герои»

Отношение 15 – «Матч\_герои»

<u>Код_матча_героя</u>	Убийства	Помощи	Смерти	ГПМ	ХПМ
Урон по героям	Урон по строениям	Добито крипов	Лечение	Код_матча	
Код_героя	<b>Код_игрока</b>				

Отношение 16 – «Игроки»

<u>Код_игрока</u>	Никнейм	Страна	Роль	Код_команды
-------------------	---------	--------	------	-------------

Рисунок 28 – Отношения 15 и 16

Следующие сущности – «Матч» и «Матч\_предметы». Связь между ними – один ко многим.

Сущность – «Матч\_предметы»

<u>Код_матч_предметы</u>	Время покупки
--------------------------	---------------

Сущность – «Матчи»

<u>Код_матча</u>	Результат	Убийств сил света	Убийств сил тьмы	Длительность
Код_команды1	Код_команды2	Код_турнира		

Рисунок 29 – Связь «Матч\_предметы – Матчи»

Отношение 17 – «Матч\_предметы»

<u>Код_матч_предметы</u>	Время покупки	<b>Код_матча</b>
--------------------------	---------------	------------------

Отношение 18 – «Матчи»

<u>Код_матча</u>	Результат	Убийств сил света	Убийств сил тьмы	Длитель- ность
Код_команды1	Код_команды2	Код_турнира		

Рисунок 30 – Отношения 17 и 18

Следующие сущности – «Герои» и «Матч\_предметы». Связь между ними – один ко многим.

Сущность – «Матч\_предметы»

<u>Код_матч_предметы</u>	Время покупки	Код_матча
--------------------------	---------------	-----------

Сущность – «Герои»

<u>Код_героя</u>	Имя_героя	Способность1	Способность2
Способность3	Способность4		

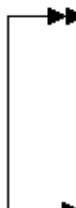


Рисунок 31 – Связь «Герои – Матч\_предметы»

Отношение 19 – «Матч\_предметы»

<u>Код_матч_предметы</u>	Время покупки	Код_матча	<b>Код_героя</b>
--------------------------	---------------	-----------	------------------

Отношение 20 – «Матч»

<u>Код_героя</u>	Имя_героя	Способность1	Способность2
Способность3	Способность4		

Рисунок 32 – Отношения 19 и 20

Следующие сущности – «Игроки» и «Матч\_предметы». Связь между ними – один ко многим.

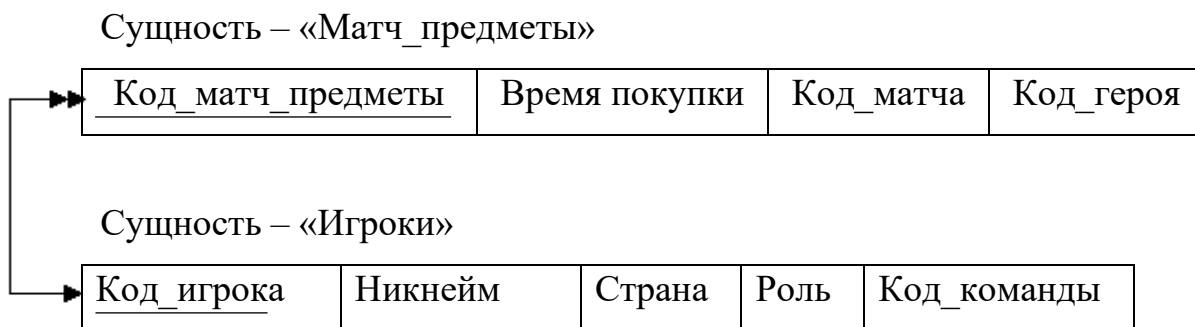


Рисунок 33 – Связь «Игроки – Матч\_предметы»

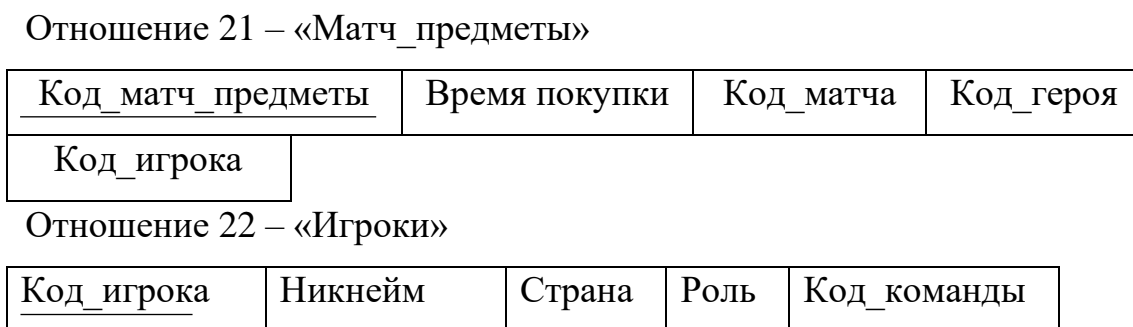


Рисунок 34 – Отношения 21 и 22

Следующие сущности – «Предметы» и «Матч\_предметы». Связь между ними – один ко многим.

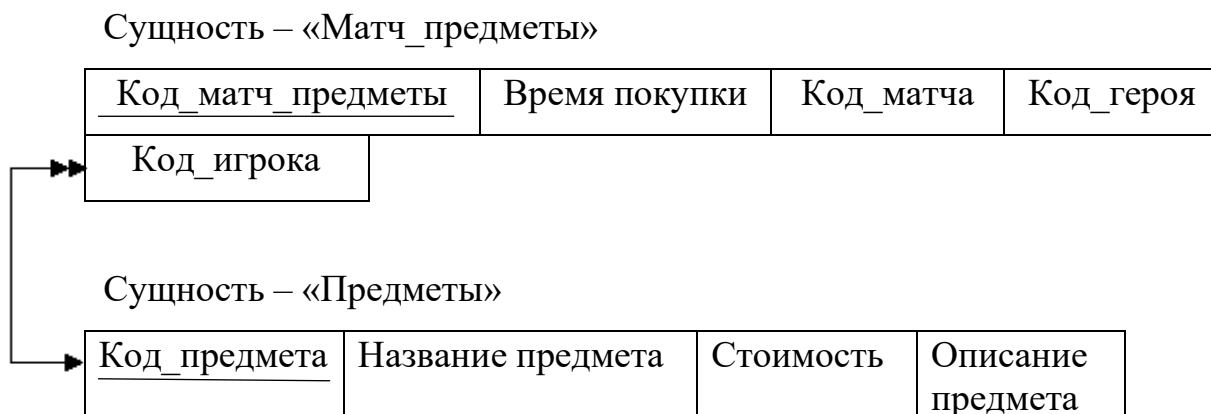


Рисунок 35 – Связь «Предметы – Матч\_предметы»

Отношение 23 – «Матч\_предметы»

<u>Код_матч_предметы</u>	Время покупки	Код_матча	Код_героя
Код_игрока	<b>Код_предмета</b>		

Отношение 24 – «Игроки»

<u>Код_предмета</u>	Название предмета	Стоимость	Описание предмета
---------------------	-------------------	-----------	-------------------

Рисунок 36 – Отношения 23 и 24

Полученные отношения необходимо проверить на соответствие нормальным формам. Все восемь отношений находятся в 1НФ, т.к. значения атрибутов не являются повторяющейся группой или множеством, следовательно, они атомарные.

Отношение «Команды» находится в 2НФ, поскольку оно находится в 1НФ, и каждый не ключевой атрибут полно и функционально зависит от ключа. На рисунке 37 изображена диаграмма функциональных зависимостей отношения «Команды».

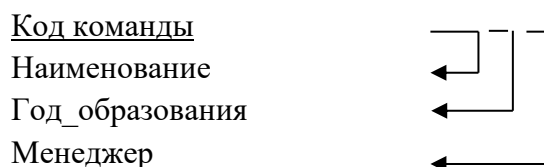


Рисунок 37 – Функциональные зависимости сущности «Команды»

Отношение «Игроки» находится в 2НФ, поскольку оно находится в 1НФ, и каждый не ключевой атрибут полно и функционально зависит от ключа. На рисунке 38 изображена диаграмма функциональных зависимостей отношения «Игроки».

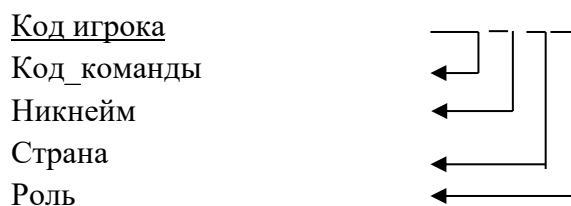


Рисунок 38 – Функциональные зависимости сущности «Игроки»

Отношение «Герои» находится в 2НФ, поскольку оно находится в 1НФ, и каждый не ключевой атрибут полно и функционально зависит от ключа. На рисунке 39 изображена диаграмма функциональных зависимостей отношения «Герои».

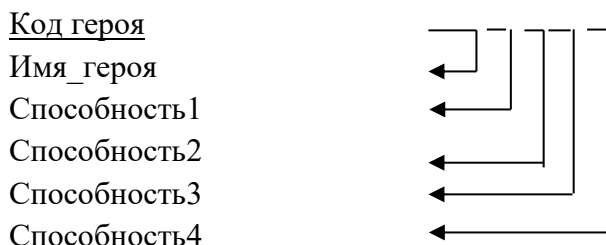


Рисунок 39 – Функциональные зависимости сущности «Герои»

Отношение «Предметы» находится в 2НФ, поскольку оно находится в 1НФ, и каждый не ключевой атрибут полно и функционально зависит от ключа. На рисунке 40 изображена диаграмма функциональных зависимостей отношения «Предметы».

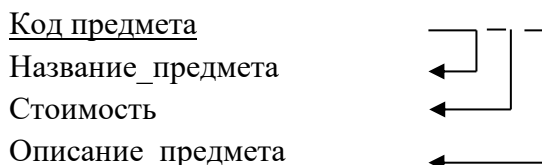


Рисунок 40 – Функциональные зависимости сущности «Предметы»

Отношение «Турниры» находится в 2НФ, поскольку оно находится в 1НФ, и каждый не ключевой атрибут полно и функционально зависит от ключа. На рисунке 41 изображена диаграмма функциональных зависимостей отношения «Турниры».

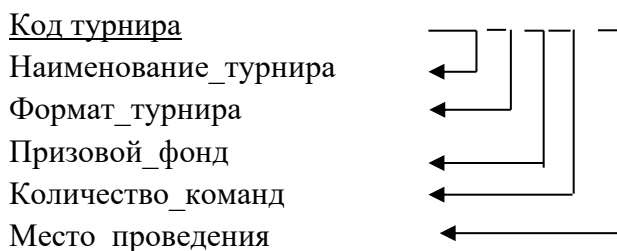


Рисунок 41 – Функциональные зависимости сущности «Турниры»

Отношение «Турниры\_команды» находится в 2НФ, поскольку оно находится в 1НФ, и каждый не ключевой атрибут полно и функционально зависит от ключа. На рисунке 42 изображена диаграмма функциональных зависимостей отношения «Турниры\_команды».

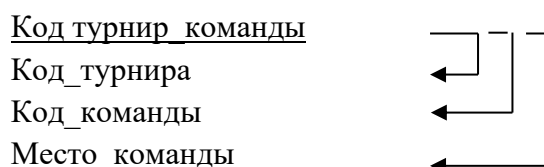


Рисунок 42 – Функциональные зависимости сущности «Турниры\_команды»

Отношение «Матчи» находится в 2НФ, поскольку оно находится в 1НФ, и каждый не ключевой атрибут полно и функционально зависит от ключа. На рисунке 43 изображена диаграмма функциональных зависимостей отношения «Матчи».

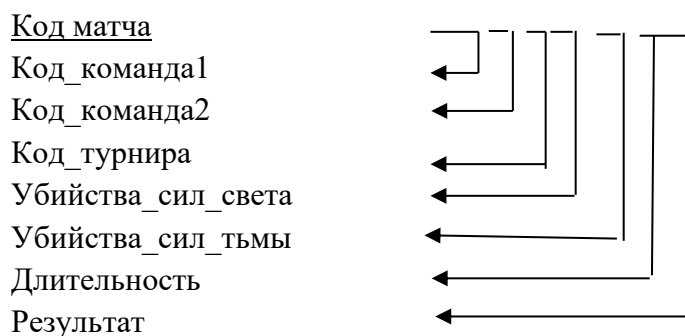


Рисунок 43 – Функциональные зависимости сущности «Матчи»

Отношение «Матч\_герои» находится в 2НФ, поскольку оно находится в 1НФ, и каждый не ключевой атрибут полно и функционально зависит от ключа. На рисунке 44 изображена диаграмма функциональных зависимостей отношения «Матч\_герои».

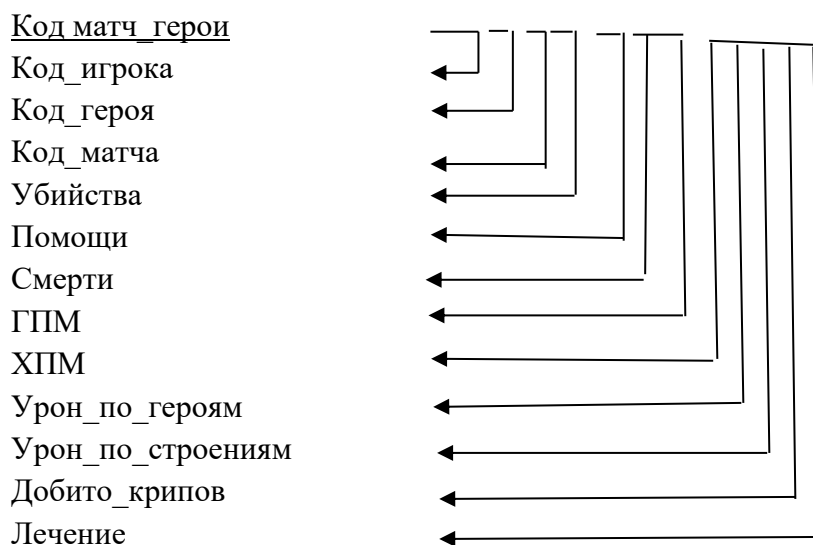


Рисунок 44 – Функциональные зависимости сущности «Матч\_герои»

Отношение «Матч\_предметы» находится в 2НФ, поскольку оно находится в 1НФ, и каждый не ключевой атрибут полно и функционально зависит от ключа. На рисунке 45 изображена диаграмма функциональных зависимостей отношения «Матч\_предметы».

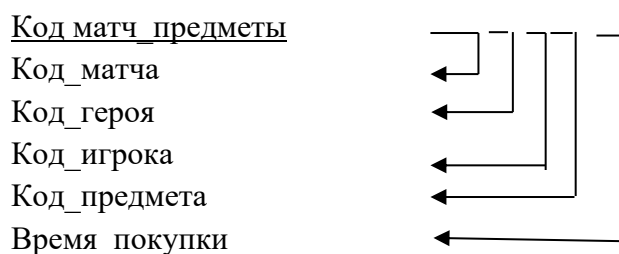


Рисунок 45 – Функциональные зависимости сущности «Матч\_предметы»

Отношения находятся в третьей нормальной форме, так как они соответствуют второй нормальной форме и все атрибуты, которые не являются ключевыми, не имеют транзитивной зависимости от ключевых атрибутов.

### 3.2.3 Физическое проектирование

Используя представленные на рисунках 37-45 функциональные зависимости, можно составить физическую модель базы данных. Составление физической модели проведено в соответствии с типами данных, используемых в СУБД.



Результат физического представления отношений представлен в таблицах 12-20.

Таблица 12 – Физическое представление отношения «Команды»

Название атрибута	Тип данных	Диапазон значений	Допустимость NULL	Индексация
<u>Код_команды</u>	Integer	>0	-	Primary Key
Наименование	VAR-CHAR(50)	-	-	-
Год_образования	DATE	-	-	-
Менеджер	VAR-CHAR(50)	-	-	-

Таблица 13 – Физическое представление отношения «Игроки»

Название атрибута	Тип данных	Диапазон значений	Допустимость NULL	Индексация
<u>Код_игрока</u>	Integer	>0	-	Primary Key
Код_команды	Integer	-	-	Foreign Key
Никнейм	VAR-CHAR(50)	-	-	-
Страна	VAR-CHAR(50)	-	-	-
Роль	VAR-CHAR(50)	-	-	-

Таблица 14 – Физическое представление отношения «Герои»

Название атрибута	Тип данных	Диапазон значений	Допустимость NULL	Индексация
1	2	3	4	5
<u>Код_героя</u>	Integer	>0	-	Primary Key
Имя_героя	VAR-CHAR(50)	-	-	-
Способность1	VAR-CHAR(50)	-	-	-
Способность2	VAR-CHAR(50)	-	-	-

1	2	3	4	5
Способность3	VAR-CHAR(50)	-	-	-
Способность4	VAR-CHAR(50)	-	-	-

Таблица 15 – Физическое представление отношения «Предметы»

Название атрибута	Тип данных	Диапазон значений	Допустимость NULL	Индексация
<u>Код_предмета</u>	Integer	>0	-	Primary Key
Название предмета	VAR-CHAR(50)	-	-	-
Стоимость предмета	Integer	-	-	-
Описание предмета	VAR-CHAR(150)	-	-	-

Таблица 16 – Физическое представление отношения «Турниры»

Название атрибута	Тип данных	Диапазон значений	Допустимость NULL	Индексация
<u>Код_турнира</u>	Integer	>0	-	Primary Key
Наименование турнира	VAR-CHAR(50)	-	-	-
Формат_турнира	VAR-CHAR(50)	-	-	-
Призовой_фонд	VAR-CHAR(50)	-	-	-
Количество_команд	Integer	-	-	-
Место_проведения	VAR-CHAR(50)	-	-	-

Таблица 17 – Физическое представление отношения «Турниры\_команды»

Название атрибута	Тип данных	Диапазон значений	Допустимость NULL	Индексация
1	2	3	4	5

1	2	3	4	5
<u>Код_турниры_команды</u>	Integer	>0	-	Primary Key
Код_турнира	Integer	-	-	Foreign Key
Код_команды	Integer	-	-	Foreign Key
Место_команды	Integer	-	-	-

Таблица 18 – Физическое представление отношения «Матчи»

Название атрибута	Тип данных	Диапазон значений	Допустимость NULL	Индексация
<u>Код_матча</u>	Integer	>0	-	Primary Key
Код_команда1	Integer	-	-	Foreign Key
Код_команда2	Integer	-	-	Foreign Key
Код_турнира	Integer	-	-	Foreign Key
Убийства сил света	Integer	-	-	-
Убийства сил тьмы	Integer	-	-	-
Длительность	TIME	-	-	-

Таблица 19 – Физическое представление отношения «Матчи\_герои»

Название атрибута	Тип данных	Диапазон значений	Допустимость NULL	Индексация
1	2	3	4	5
<u>Код_матч_герои</u>	Integer	>0	-	Primary Key
Код_игрока	Integer	-	-	Foreign Key
Код_героя	Integer	-	-	Foreign Key
Код_матча	Integer	-	-	Foreign Key

1	2	3	4	5
Убийства	Integer	-	-	-
Помощи	Integer	-	-	-
Смерти	Integer	-	-	-
ГПМ	Integer	-	-	-
ХПМ	Integer	-	-	-
Урон_по_героям	Integer	-	-	-
Урон_по_строениям	Integer	-	-	-
Добито_крипов	Integer	-	-	-
Лечение	Integer	-	-	-

Таблица 20 – Физическое представление отношения «Матчи\_предметы»

Название атрибута	Тип данных	Диапазон значений	Допустимость NULL	Индексация
<u>Код_матч_предметы</u>	Integer	>0	-	Primary Key
Код_игрока	Integer	-	-	Foreign Key
Код_героя	Integer	-	-	Foreign Key
Код_матча	Integer	-	-	Foreign Key
Код_предмета	Integer	-	-	Foreign Key
Время_покупки	TIME	-	-	-

На основе данных физических представлений построена схема физического проектирования базы данных, представленная на рисунке 46.

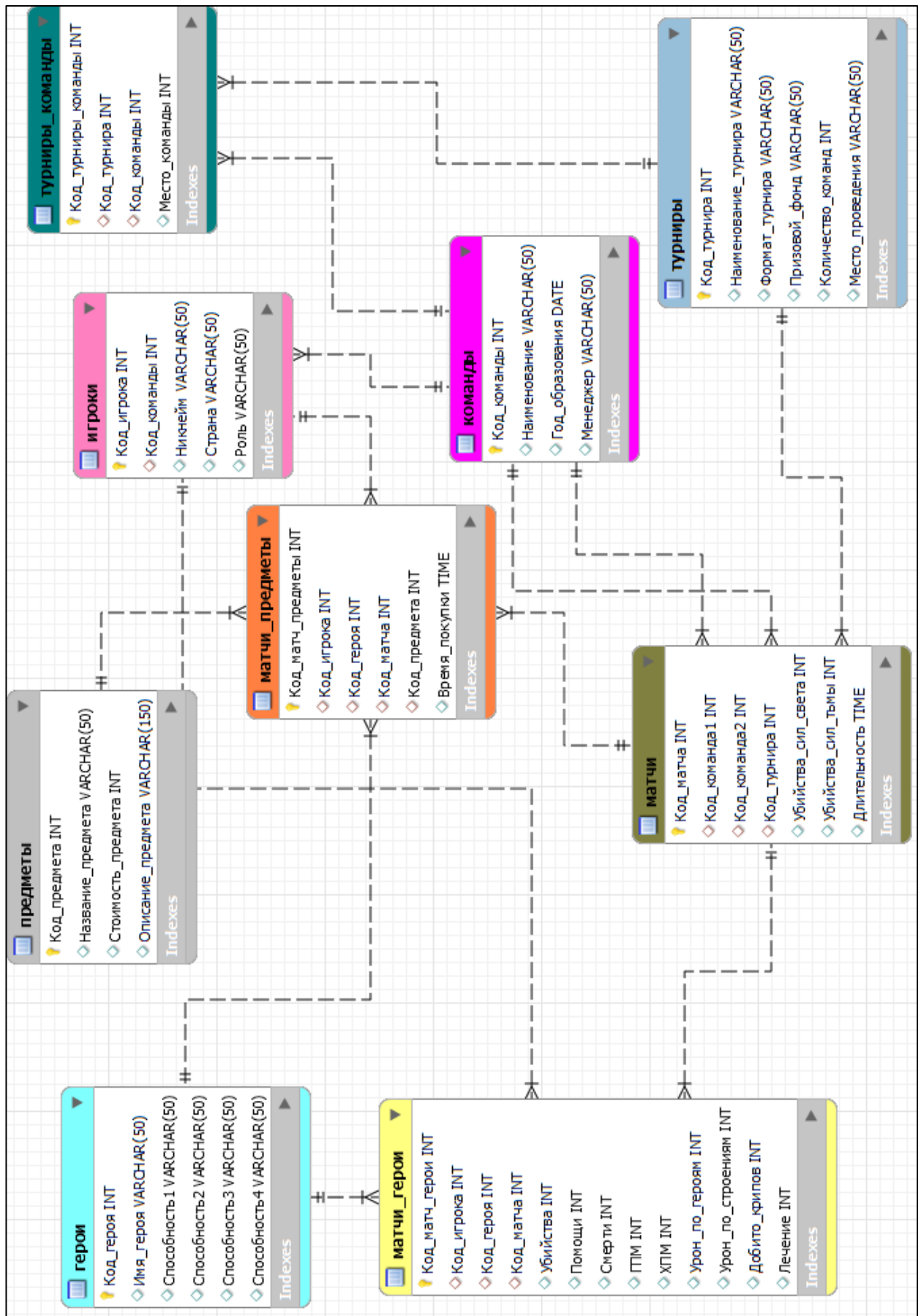


Рисунок 46 – Физическая модель БД

Спроектированная база данных позволит структурировать и организовать информацию в удобной форме, что помогает улучшить эффективность работы с данными. Она будет полезна в различных сферах, связанных с управлением и анализом информации. Некоторые возможные применения базы данных включают:

- спортивная аналитика: хранение информации о командах, игроках, героях, предметах, турнирах и матчах, это обеспечивает возможность проведения анализа данных, определения статистики, трендов и результативности команд и игроков;

- управление турнирами: хранение информации о турнирах, командах и их результатах, это позволяет эффективно планировать, организовывать и управлять турнирами, включая учет команд, расписание матчей, подсчет очков и призовых фондов;

- учет игроков и команд: возможность вести учет игроков, команд и их характеристик, это помогает в управлении составами команд и отслеживании статистики игроков;

- управление предметами и героями: хранение информации о предметах и героях, это позволяет управлять списком доступных предметов и героев, а также проводить анализ их эффективности и баланса в игре;

В целом, разработанная база данных предоставляет основу для хранения, управления и анализа данных, связанных с игровой сферой и спортивными событиями. Она помогает организовать информацию, обеспечивает удобный доступ к данным и обеспечивает возможности для аналитики и управленческих решений.

### **3.3 Разработка алгоритмов сбора данных**

Рассмотрим функционал программы-парсера для заполнения информации о героях. Программа анализирует список персонажей, публикующийся на сайте [opendota.com](http://opendota.com), а следом заполняет таблицу `heroes` базы данных `dota`.

Рассмотрим структуру таблицы `Heroes`. Она состоит из трех полей, таких как:

- id – идентификатор каждой записи в таблице;
- name – наименование героя;
- abilities – наименование способностей.

Для реализации проекта приложения-парсера используется язык программирования Python с фреймворком Scrapy.

Выбор фреймворка Scrapy обоснован его открытой архитектурой, то есть возможностью расширения функций пакета собственными модулями.

Нужно провести инспектирование страницы, чтобы определить какой селектор использовать для отбора элементов DOM-дерева страницы.

На странице героев нас интересует тэг – `<div class = "hero-grid">...</div>` (рис. 47).

```

▶ <div class="header-content-container">...</div>
▼ <div class="content-inner">
  ▶ <section style="margin-bottom: 0px">...</section>
  ▶ <div class="row-12">...</div>
  ▼ <section>
    ▶ <header>...</header>
    ▼ <footer style="padding: 0">
      ▶ <div class="hero-grid">...</div> == $0
    </footer>
  </section>
  <div class="multishot molto-grande" data-type="elo-placement" data-space="desktop_footer"></div>
  <div class="multishot molto-piccolo" data-type="elo-placement" data-space="mobile_footer"></div>
  <div class="multishot" data-type="elo-placement" data-space="footer"></div>
</div>
▶ <div class="footer-nav-primary">...</div>
▶ <div class="footer-nav-secondary">...</div>

```

Рисунок 47 – Инспектирование страницы «Героев»

Программный код для считывания информации о героях представлен на рисунке А.1.

Первым шагом программы является определение URL-адреса страницы, содержащей информацию о героях. Затем создается класс-краулер, который наследуется от класса scrapy.Spider. В нем описываются методы, позволяющие собрать нужную информацию со страницы.

В методе «parse» происходит обработка страницы. Сначала с помощью CSS-селектора находится блок с таблицей, содержащей информацию о героях.

Затем в цикле по строкам таблицы собираются данные об имени героя и списке его способностей. Данные сохраняются в экземпляре класса HeroItem.

Далее в классе Spider определен метод, который вызывается после обработки страницы. В нем происходит сохранение данных в базу данных MySQL. Для этого создается экземпляр класса, наследующего от класса scrapy.Item, в котором определены поля таблицы, соответствующие собранному данным. Затем вставляются данные в таблицу.

Таким образом, программа позволяет собрать список героев и список их способностей с сайта и сохранить их в базу данных MySQL для последующего использования в работе.

Подобный пример будет работать для каждой статичной таблицы в базе данных, таких как, «Герои» и «Предметы».

Теперь рассмотрим программу-парсер для считывания информации о прошедших матчах.

Снова произведем инспектирование страницы (рис. 48) для определения селектора.

На страницах матчей нас интересует следующий тэг – `<div class = “team-results”>...</div>`.

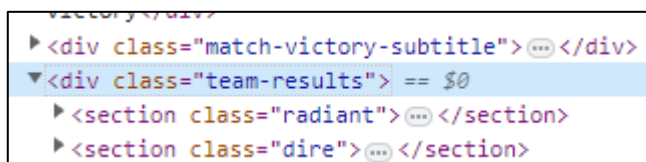


Рисунок 48 – Инспектирование страницы

Данные, которые необходимо считывать:

Название команды, никнейм игрока, убийства, смерти, помощи, добито крипов, ГПМ, ХПМ, урон по героям, урон по строениям, лечение, а также список купленных предметов (рис.49).



TSM FTX - Обзор												
ИГРОК	УР.	У	С	П	ДК/НО	NET	З/М/О/М	УРОН	ПСТР	ЛЕЧ	ПРЕДМЕТЫ	
Титан	16	2	12	2	83/1	7.6k	299/381	9.4k	231	390	26:00 9:41 12:29 14:34 30:26	
Титан	19	-	6	2	258/7	13k	418/511	10.9k	480	-	5:02 8:05 27:55 14:44 23:23 4:11 1:35	
Титан	18	2	8	1	198/5	11.1k	393/457	7.6k	1.3k	-	29:51 13:04 8:41 6:34 19:19 -1:29 0:29	
Титан	13	-	3	2	23/1	3.5k	181/265	6.7k	-	123	11:47 25:26 21:32	
Титан	19	-	8	1	299/1	15.2k	487/514	6.1k	606	-	-1:29 17:05 5:47 31:01 11:02 21:13 -1:29	
	85	4	37	8	861/15	50.4k	1.8k/2.1k	40.6k	2.6k	513		

Рисунок 49 – Послематчевая статистика стороны сил света

Программный код для считывания для получения послематчевой статистики изображен на рисунке А.2.

Сначала определен класс DotaMatchSpider, который наследуется от класса scrapy.Spider, который предоставляет основные функции для работы с веб-скрапингом. Данный класс имеет атрибут name, который устанавливает имя скрапера и используется при его запуске.

Далее, в методе start\_requests() указываются URL-адреса матчей, которые будут скрапиться. Для каждого URL-адреса создается объект scrapy.Request, который отправляется на сайт для получения ответа. После получения ответа вызывается метод parse(), который обрабатывает полученные данные.

В методе parse() происходит парсинг полученной страницы. Сначала с помощью CSS-селекторов извлекаются названия команд и никнеймы игроков. Затем извлекаются данные об убийствах, смертях, помощи, добитых крипах, ГПМ, ХПМ, уроне по героям, уроне по строениям и лечении для каждого игрока. Для этого в цикле обходятся все игроки на странице и для каждого из них извлекаются соответствующие данные с помощью CSS-селекторов. Полученные данные сохраняются в словаре, который затем добавляется в список player\_stats.

Затем извлекается URL-адрес следующего матча с помощью XPath-выражения, и если он существует, то создается новый объект scrapy.Request для скрапинга следующего матча. После этого данный объект передается в метод callback

для обработки ответа. Если следующий матч не найден, то скрапер завершает свою работу.

Таким образом, данный скрапер позволяет получать данные о матчах в игре Dota 2. Следом, данные о прошедших матчах записываются в таблицу «Матчи».

Но перед записью данных об активных составах команд в базу данных необходимо провести очистку данных, чтобы эти данные были актуальны. Производить это необходимо с помощью алгоритма обработки MapReduce (рис. А.3):

- на этапе Map нужно прочитать данные из файла с информацией о матчах и преобразовать их в пары ключ-значение, где ключ – это команда, а значение – список всех игроков, которые когда-либо играли в этой команде, для этого можно использовать функцию mapper, которая будет считывать данные и формировать пары ключ-значение;

- на этапе Shuffle пары ключ-значение из Map будут сгруппированы по ключу, то есть по команде, для этого не нужно производить каких-либо дополнительных действий, это делает сам фреймворк MapReduce;

- на этапе Reduce нужно произвести обработку каждой группы, т.е. для каждой команды нужно определить, какие игроки являются активными, а какие нет, для этого нужно проанализировать последний матч каждой команды и оставить только тех игроков, которые участвовали в этом матче, здесь необходимо использовать функцию reducer, которая будет принимать пары ключ-значение и фильтровать игроков;

- после выполнения Reduce необходимо записать результаты в файл .json формата, для этого можно использовать стандартную библиотеку Python json и функцию json.dump().

После очистки запишем информацию в базу данных.

### **3.4 Разработка алгоритмов для обработки данных**

Большинство алгоритмов машинного обучения требуют, чтобы все данные были переведены в числовое значение, следовательно на этом этапе необходимо их подготовить. Для этого создаётся датасет-файл.

Создание датасета – это процесс, при котором информация извлекается из базы данных и преобразуется ее в формат, понятный для модели машинного обучения.

Для модели создается датасет, который будет содержать информацию о каждом матче, в котором участвовал герой. Для этого необходимо выбрать все матчи, в которых участвовал герой, и для каждого матча извлечь следующую информацию: идентификатор матча, героя, команду, в которой он играл, выбранных героев противника, результат матча (победа или поражение).

Для извлечения данных используется язык запросов к базе данных. Полученные данные из базы будут сохраняться в формате CSV или любом другом формате, удобном для работы с моделями машинного обучения.

Примерный вид датасета представлена в таблице 21.

Таблица 21 – Датасет для обучения модели предложения контрпика

Идентификатор матча	Герой	Герои противника	Результат матча
1	Axe	Dazzle, Sniper	Победа
2	Axe	Lina, Pudge	Поражение
3	Axe	Mirana, Windranger	Победа
4	Axe	Juggernaut, Lion	Поражение
...	...	...	...

Преобразование данных может включать в себя различные операции, например, преобразование категориальных признаков в числовые, нормализацию числовых признаков, удаление выбросов и т.д. В данном случае будут преобразовываться категориальные признаки, такие как герои и команды, в числовые, например, с помощью метода One-Hot Encoding. Также необходимо нормализовать числовые признаки, например, результаты матчей, чтобы они находились в диапазоне от 0 до 1.

Для обучения модели, необходимо подготовить данные и разделить их на три выборки: обучающую, валидационную и тестовую.

В качестве исходных данных используется предварительно подготовленный датасет.

Перед тем, как разделить данные на выборки, необходимо перемешать их случайным образом. Это позволит избежать ситуации, когда данные разделены на выборки с определенным порядком и они могут содержать некоторую зависимость.

Затем данные разделяются на обучающую, валидационную и тестовую выборки в соотношении 70/15/15.

Обучающая выборка будет использоваться для обучения модели на основе имеющихся данных. Валидационная выборка будет использоваться для подбора наилучших параметров модели и для оценки ее качества. И, наконец, тестовая выборка будет использоваться для окончательной проверки качества модели.

При разделении данных на выборки необходимо учитывать, что каждый матч должен попасть только в одну выборку, чтобы избежать переобучения и увеличить обобщающую способность модели [51].

Для разделения исходного датасета (рис. А.4) на выборки используется функция `train_test_split()` библиотеки `scikit-learn`. При вызове этой функции указываются исходные данные (матрица признаков и вектор целевых переменных), а также параметры `test_size` и `random_state`.

Параметр `test_size` определяет долю данных, которые будут использоваться для тестирования модели. Обычно это значение составляет 20-30% от исходного датасета. Параметр `random_state` устанавливает начальное значение генератора случайных чисел, что позволяет воспроизводить результаты при повторных запусках кода.

### **3.5 Разработка алгоритма для создания модели машинного обучения**

В зависимости от задачи и типа данных, который используется для обучения, подходящий алгоритм для обучения модели может различаться. Однако, для задачи классификации, хорошо подходят такие алгоритмы, как логистическая

регрессия, решающие деревья, случайный лес, градиентный бустинг и нейронные сети.

В данном случае, можно использовать алгоритмы машинного обучения, такие как градиентный бустинг и случайный лес. Оба этих алгоритма хорошо подходят для решения задачи классификации и могут работать с различными типами данных. Однако, для данной задачи лучше использовать градиентный бустинг, так как он обычно имеет лучшую точность предсказаний, чем случайный лес, при этом не переобучается на обучающих данных.

Градиентный бустинг (Gradient Boosting) – метод машинного обучения, который строит предсказательную модель в виде композиции функций (ансамбля), используя градиентный спуск для минимизации ошибки [22].

Для обучения модели можно использовать библиотеки машинного обучения, такие как Scikit-learn или XGBoost, которые имеют встроенную реализацию градиентного бустинга.

Код для обучения модели представлен на рисунке А.5.

Этот код использует библиотеку XGBoost для создания модели градиентного бустинга. Сначала данные разделяются на обучающую и тестовую выборки с помощью функции `train_test_split`. Затем создается объект модели `XGBClassifier()`. Модель обучается на обучающей выборке с помощью метода `fit`. После обучения модель делает предсказания на тестовой выборке с помощью метода `predict`. Наконец, качество модели оценивается с помощью метрики точности (`accuracy_score`) и выводится на экран.

Обучение модели градиентного бустинга происходит путем последовательного добавления деревьев к уже имеющейся модели. Каждое новое дерево обучается на ошибках, сделанных предыдущими деревьями, с целью уменьшить эти ошибки. Градиентный бустинг является итеративным алгоритмом, поэтому количество деревьев (итераций) должно быть достаточным для достижения желаемой точности предсказаний. В данном случае, используется метрика `accuracy` для оценки качества модели.

Для реализации предложения контрпика используется уже обученная модель, ей на вход передается выбранный герой. Затем модель производит предсказание и выдает рейтинг всех остальных героев, среди которых пользователь сможет выбрать наиболее подходящего для борьбы с выбранным героем.

Код для реализации этой функции указан на рисунке А.6.

Функция `recommend_heroes` начинается с создания дата фрейма `df_hero`, который содержит информацию только о матчах, в которых был выбранный герой. Затем создается список `enemy_heroes` всех героев, против которых играл выбранный герой. Далее создается дата фрейм `df_enemies`, содержащий информацию о матчах, в которых участвовали выбранные герои противника.

Далее создается матрица признаков  $X$  и вектор целей  $Y$ , которые будут использоваться для обучения модели. Вектор целей  $Y$  содержит информацию о том, кто победил в каждом матче, а матрица признаков  $X$  содержит информацию обо всех героях, кроме выбранного пользователем.

Затем производится предсказание вероятности победы для каждого героя, кроме выбранного пользователем. Полученные результаты сортируются в порядке убывания рейтинга, и функция возвращает список наилучших героев для выбора пользователем.

Код возвращает список лучших героев для противостояния выбранному герою. Для вывода этого списка используется `return top_heroes['hero'].tolist()`. Кроме того, можно передать необязательный параметр `num_recommendations`, который определяет количество наилучших героев, которые будут выведены.

Код для реализации функции дополнения пика представлен на рисунке А.7.

Функция `recommend_plushero` используется для рекомендации следующего героя для выбора, который будет лучше всего подходить на основе обученной модели.

Первым шагом функция создает дата фрейм `df_heroes`, содержащий информацию только о матчах, в которых принимали участие выбранные герои. Затем

она создает список всех героев, за которых играли противники выбранных героев, и создает датафрейм `df_enemies`.

Далее функция создает матрицу признаков и вектор целей на основе `df_enemies` и обученной модели `model`. Затем производится предсказание вероятностей победы для всех героев, кроме выбранных, и создается дата фрейм `hero_ratings`, содержащий информацию о героях и их рейтинге.

Далее герои сортируются по рейтингу в порядке убывания, и выбирается лучший герой, которому соответствует самый высокий рейтинг. Функция выводит на экран список лучших героев для выбора пользователем.

### **3.6 Выбор архитектуры**

Для реализации программного продукта была выбрана клиент-серверная архитектура (рис. 50).

Клиент-серверная архитектура является широко распространенным подходом, который позволяет эффективно организовать взаимодействие между клиентскими и серверными компонентами приложения. Некоторые причины выбора клиент-серверной архитектуры включают:

- распределение задач: клиентская часть приложения отвечает за представление данных пользователю и обработку пользовательских действий, в то время как серверная часть отвечает за обработку бизнес-логики и хранение данных, такое разделение позволяет более гибко масштабировать систему и распределить нагрузку;
- централизованное управление данными: клиенты обращаются к серверу для получения данных и отправки запросов на обновление информации, это обеспечивает централизованное хранение и управление данными, что улучшает их целостность и безопасность;
- повышение производительности: сервер может выполнять сложные вычисления и операции, что позволяет снизить нагрузку на клиентскую сторону и обеспечить более быструю обработку данных [18].

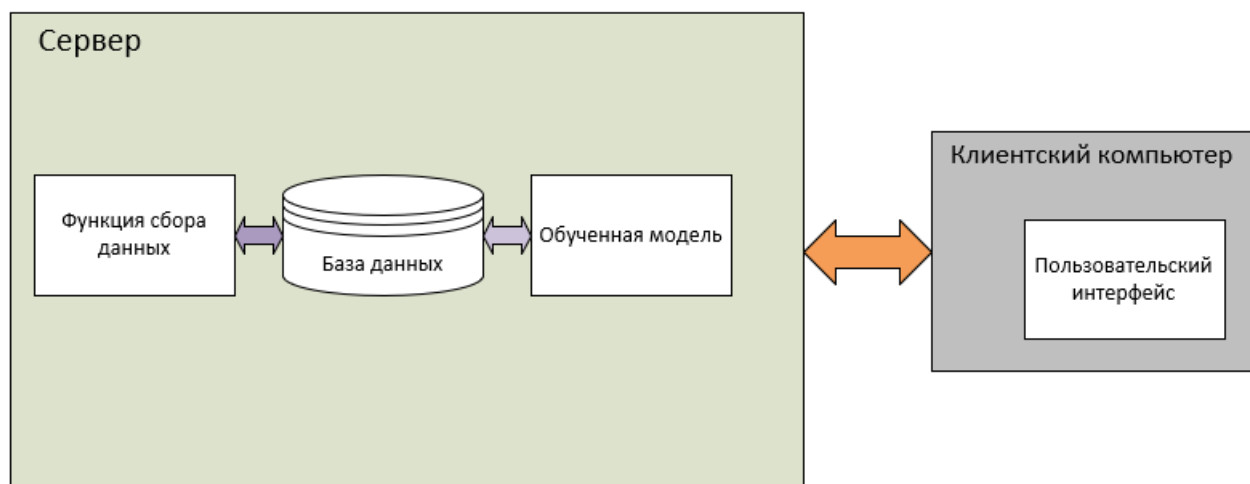


Рисунок 50 – Схема архитектуры программного продукта

Для реализации клиент-серверной архитектуры используются соответствующие технологии и протоколы. В данном проекте используется язык программирования Python для создания клиентской части приложения и сервера. Для обмена данными между клиентом и сервером используется протокол HTTP, который обеспечивает надежную и безопасную передачу информации.

В результате выбора и реализации клиент-серверной архитектуры создастся эффективное и масштабируемое приложение, обеспечивающее удобное взаимодействие с данными и отвечающее требованиям проекта.

### 3.7 Разработка серверной части

Для разработки серверной части выбран фреймворк Django, так как в настоящее время он является наиболее популярным среди разработчиков. Django – свободный фреймворк для создания веб-приложений на языке Python, использующий шаблон проектирования, аналогичный шаблону MVC.

В Django обработчики URL указываются явно при помощи регулярных выражений, в отличие от некоторых других фреймворков, где они создаются автоматически на основе структуры моделей контроллеров. Такая явная указанность обработчиков URL позволяет точно определить, какой код должен быть выполнен для каждого URL.

С использованием Django можно создавать динамические веб-приложения в короткие сроки. Фреймворк спроектирован таким образом, чтобы



разработчики могли сосредоточиться на решении более значимых задач, не отвлекаясь на рутинные вещи, такие как аутентификация и авторизация пользователей, настройка работы с базой данных и т. д. Django предоставляет общепотребительные шаблоны веб-разработки высокого уровня абстракции, инструменты для быстрого выполнения часто встречающихся задач программирования и четкие соглашения о способах решения проблем.

Архитектура Django аналогична шаблону MVC, где уровень контроллера классической модели MVC соответствует уровню "View" в Django, а презентационная логика представлений реализуется через уровень "Template". Из-за такой организации, архитектуру Django иногда называют MTV – "Model-Template-View".

При поступлении запроса на определенный URL, сервисы, построенные на Django, обращаются к файлу привязки URL, который имеет стандартное название `urls.py`. Этот файл можно рассматривать как таблицу содержимого сервера, определяющую соответствие между URL и функциями представлений (из модуля `views.py`), которые должны быть вызваны для обработки этих URL. Таким образом, для каждого запрошенного URL указывается, какой код необходимо выполнить.

Модуль `views.py` содержит функции представлений, которые отвечают за формирование HTTP-ответа. Каждое представление представляет собой функцию на языке Python. При формировании ответа представления используют HTML-шаблоны из папки с шаблонами, модели из файла `models.py`, формы из файла `forms.py` и данные из базы данных, к которой сервер подключается.

### **3.8 Разработка клиентской части**

Для разработки клиентской части информационной системы выбраны основные технологии для WindowsForm-приложений, выбранная среда разработки Visual Studio предоставляет их полной мере.

Главная форма (рис.51) содержит 6 основных разделов:

- Teams содержит информацию о киберспортивной команде;
- Players содержит информацию о выбранном киберспортсмене;

- Heroes содержит информацию о выбранном пользователем персонаже;
- Counter-pick на основе выбранного пользователем персонажа предлагает лучшие решения;
- Plus-pick на основе выбранных персонажей предлагает пользователю лучшие дополнения для набора персонажей;
- Setting содержит параметры для настройки программы.

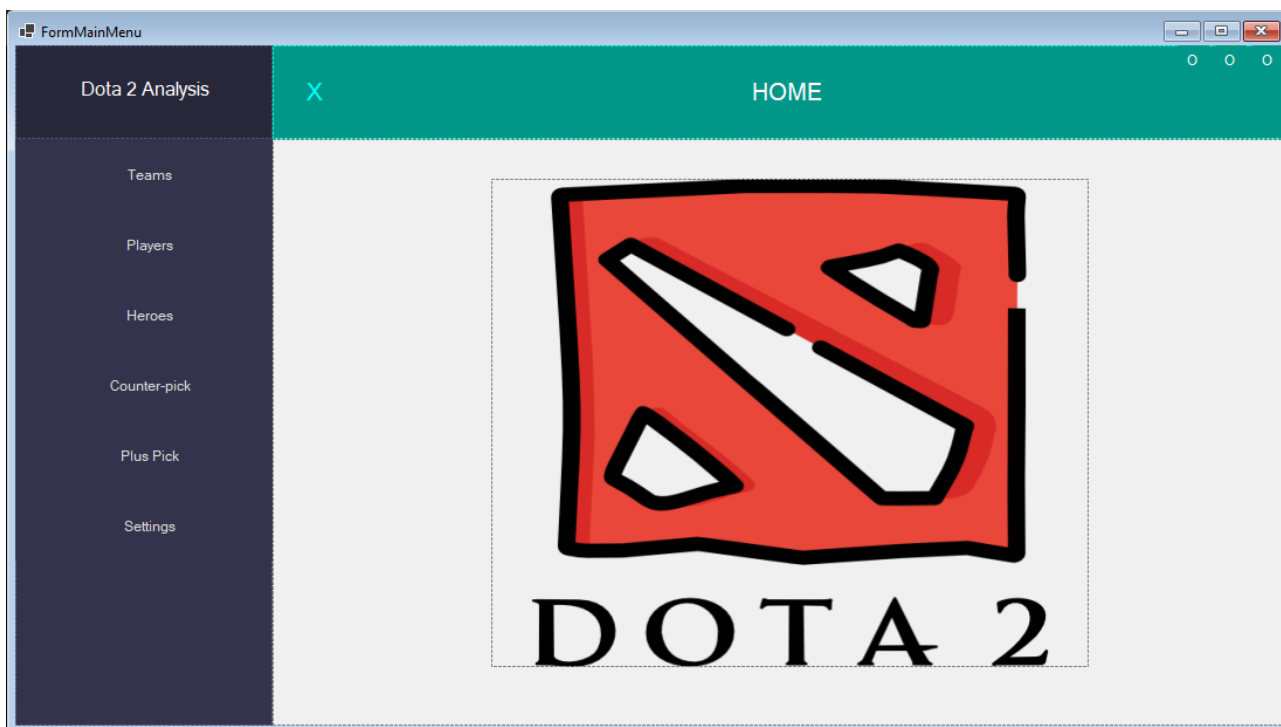


Рисунок 51 – Главное меню программы

Главная форма оборудована кнопкой сброса, которая возвращает пользователя с дочерней формы на главный экран, также присутствуют 3 кнопки для управления окном программы (свернуть, развернуть и закрыть).

Форма справки о команде (рис. 52) оборудована списком, в котором пользователю предлагается выбор команды, на основе выбранной команды, пользователю на экран будет произведен вывод логотипа команды, списка игроков команды с фотографиями, а также указанием роли и никнейма, кроме этого пользователь получит информацию о 5 последних матчах выбранной команды, с указанием логотипа и название команды оппонента.

The screenshot shows a window titled "Teams" with a standard Windows-style title bar. Inside the window, there is a form with the following elements:

- A dropdown menu at the top left.
- A "Select Team" button to the right of the dropdown.
- A large dashed rectangular box containing a smaller dashed box labeled "NameTeam".
- To the right of the "NameTeam" box, there are five rows of input fields:
  - ScoreFirst  TeamFirst
  - ScoreSecond  TeamSecond
  - ScoreThird  TeamThird
  - ScoreFourth  TeamFourth
  - ScoreFifth  TeamFifth
- Below these fields, there are five dashed rectangular boxes representing role selection:
  - Carry (NicknameCarry)
  - MidLane (NicknameMidLane)
  - OffLane (NicknameOffLane)
  - SemiSupport (NicknameSemiSupport)
  - Support (NicknameSupport)

Рисунок 52 – Форма справки о команде

Форма справка об игроке (рис. 53) оборудована текстовым полем для ввода никнейма игрока, на основе введенного никнейма пользователь получает информацию об игроке и его 5 лучших игровых персонажах, вместе со списком персонажей, пользователь получает статистическую информацию об успешности игрока при игре на данных персонажах, такую как количество игр, процент побед, соотношение убийств, помощи и смертей, количество заработанного золота в минуту, количество заработанного опыта в минуту и среднее количество нанесенного урона.

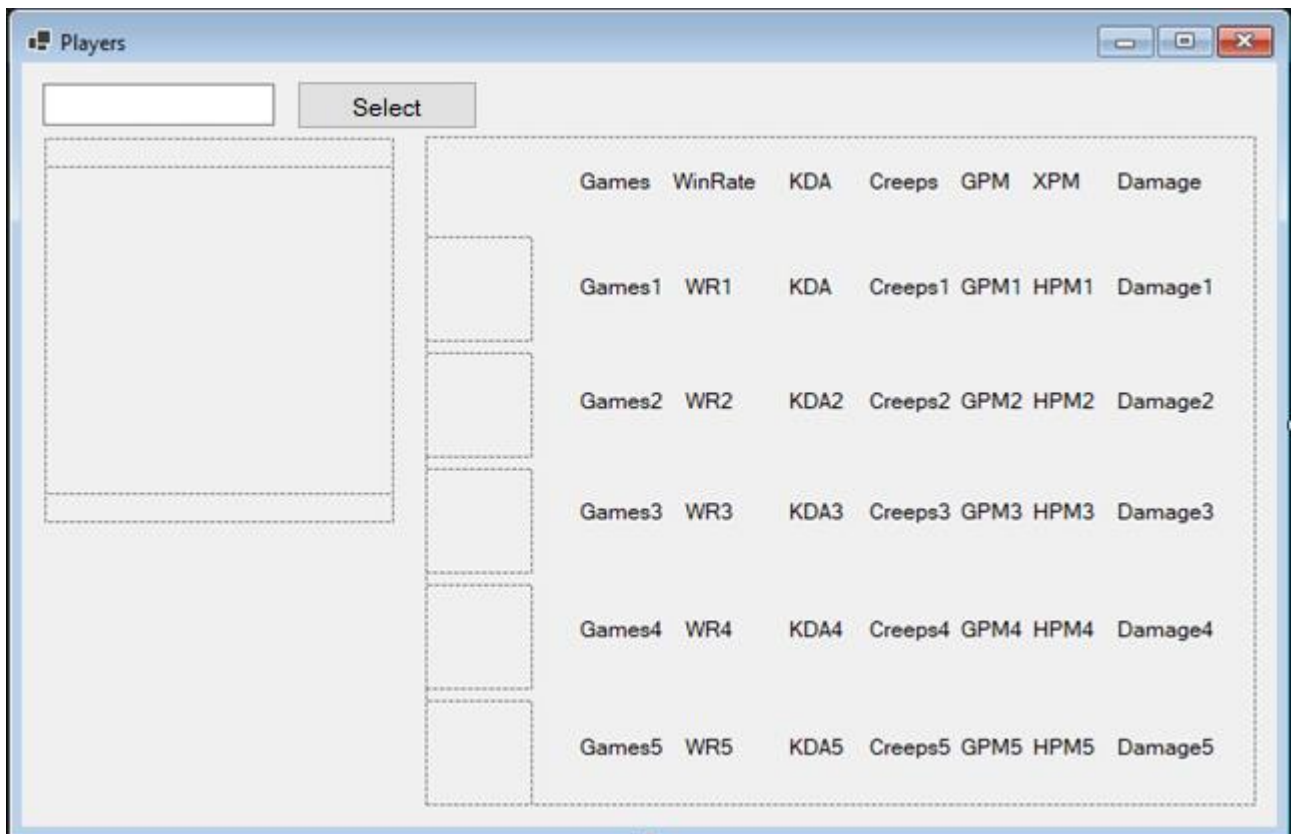


Рисунок 53 – Форма справка об игроке

На форме справка о героях (рис. 54) присутствует список для выбора персонажей, на основе выбранного пользователем персонажа на экран выводится статистика игр данного персонажа. На форму выводится картинка героя, панель со средними показателями при игре на данном персонаже, такими как процент выбора героя, процент побед, среднее соотношение убийств, помощи и смертей, средний показатель заработанного золота в минуту и средний показатель заработанного опыта в минуту. Также на форме присутствует панель с наиболее используемыми предметами для выбранного персонажа. В информацию о предметах входят такие данные, как стоимость предмета, количество матчей, количество побед и процент побед.

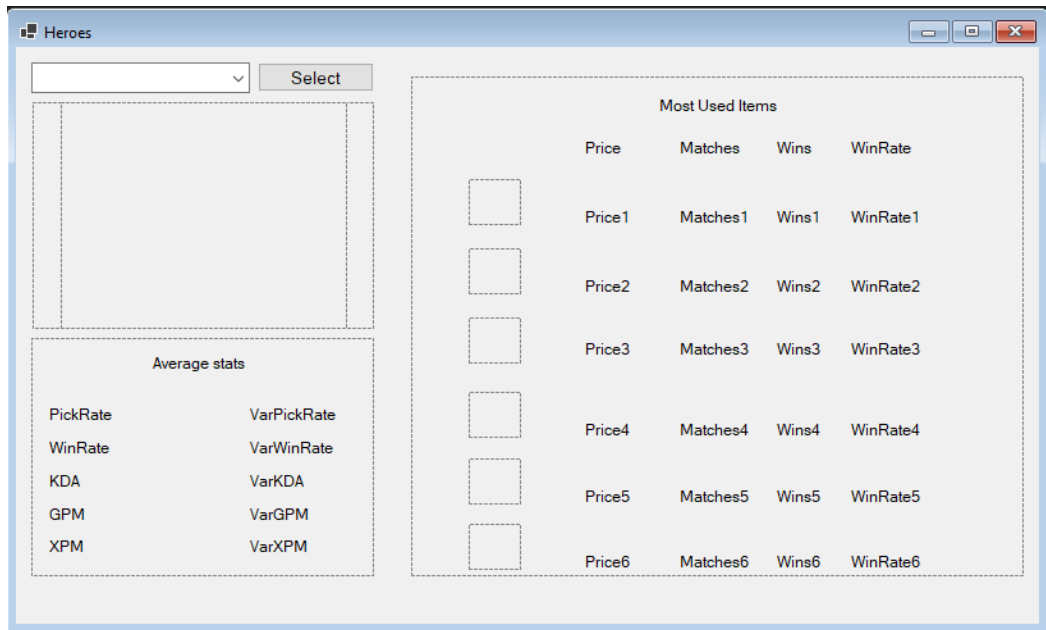


Рисунок 54 – Форма справки о героях

Форма предложения контрпики (рис. 55) содержит окно для выбора пользователем игрового персонажа, на основе выбранного пользователем персонажа программа предоставляет пользователю список из пяти героев с указанием количество сыгранных матчей, процента побед, а также соотношения убийств, помощи и смертей против указанного персонажа, отсортированный по убыванию процента побед.

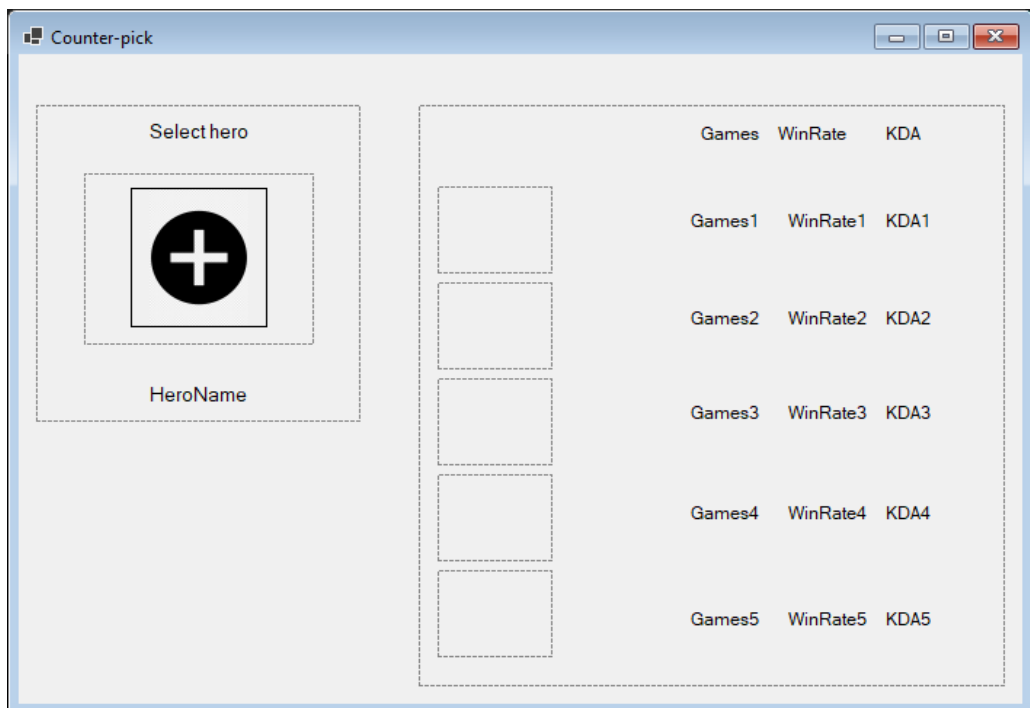


Рисунок 55 – Форма предложения контрпики

На форме предложения дополнения пика (рис. 56) расположены четыре окна для выбора игровых персонажей, кнопка для осуществления начала анализа, а также панель, на которой располагается список из трех игровых персонажей, предложенных программой, отсортированный по убыванию показателя процента побед.

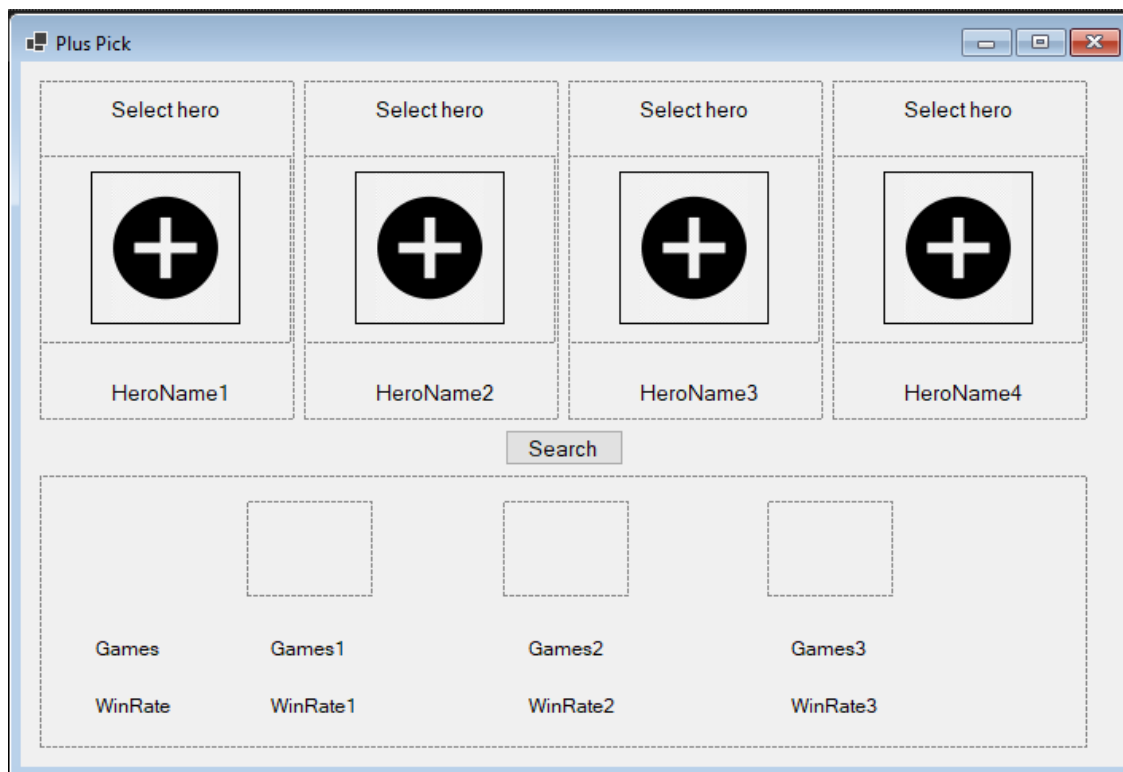


Рисунок 56 – Форма предложения дополнения пика

Также в программе предусмотрен оффлайн режим, при котором присутствует возможность производить работу с ресурсами программы без непосредственного подключения к серверу. Для подключения оффлайн режима пользователю необходимо перейти на вкладку Settings (рис. 57) и переключить программу в «Offline mode». Для успешного продолжения работы с программой актуальная база данных будет скачана на машину пользователя.

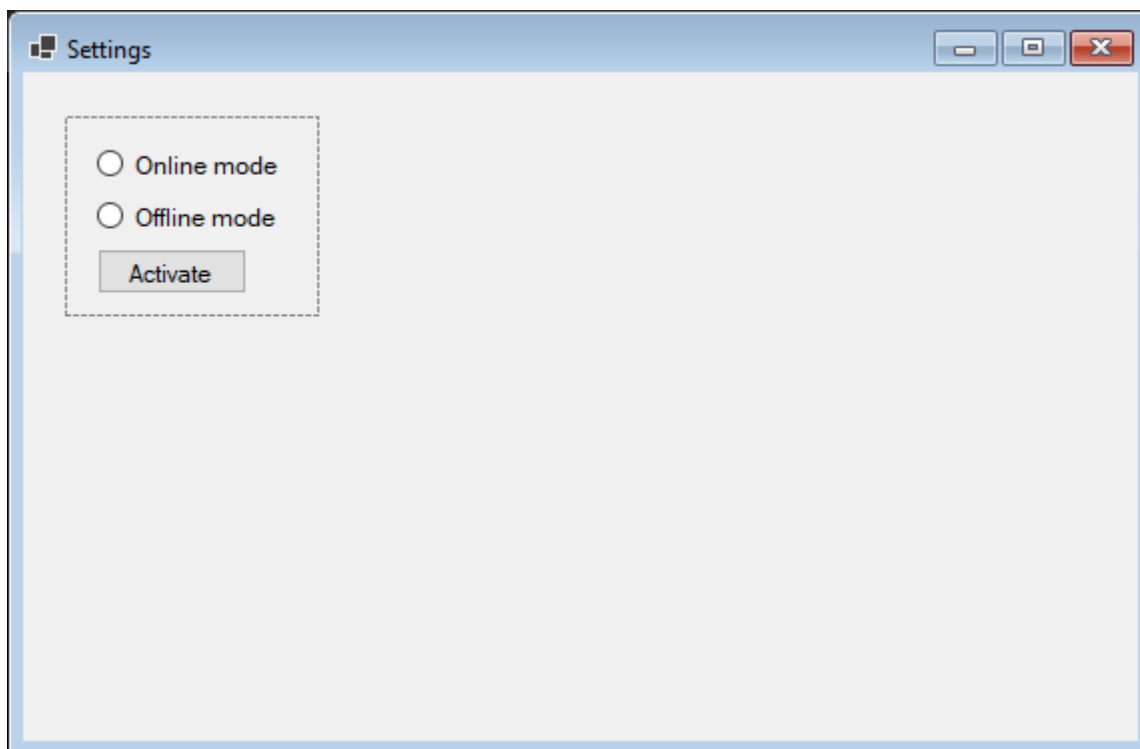


Рисунок 57 – Форма с параметрами

Также на клиентской части приложения предусмотрена проверка правильности вводимых данных. При условии, если пользователь введет не весь набор данных, то он получит соответствующее оповещение.

### 3.9 Фактическое тестирование программного продукта

Для тестирования программного продукта была выбрана киберспортивная команда – Team Spirit (рис. 58). Состав команды Yatoro, TORONTOTOKYO, Collapse, Mira, Miroshka. Также получаем актуальный логотип команды, результаты предыдущих пяти встреч команды «Team Spirit», а также действующий состав команды с указанием никнеймов и игровых ролей.

Для тестирования информации об игроке был выбран Илья «Yatoro» Мулярчук (рис. 59). На основе выбранного игрока получаем его фотографию, а также список его лучших игровых персонажей (Morphling, Faceless Void, Phantom Assassin, Ursa, Anti mage), а также информацию о статистических показателях игрока на данных героях.

Для тестирования информации о героях был выбран игровой персонаж – Phantom Assassin (рис. 60). Программа предоставляет средние статистические

показатели данного героя, а также список наиболее часто покупаемых предметов для данного героя (Black King Bar, Power Treads, Desolator, Battle Fury, Aghanim Shard, Skull Basher).

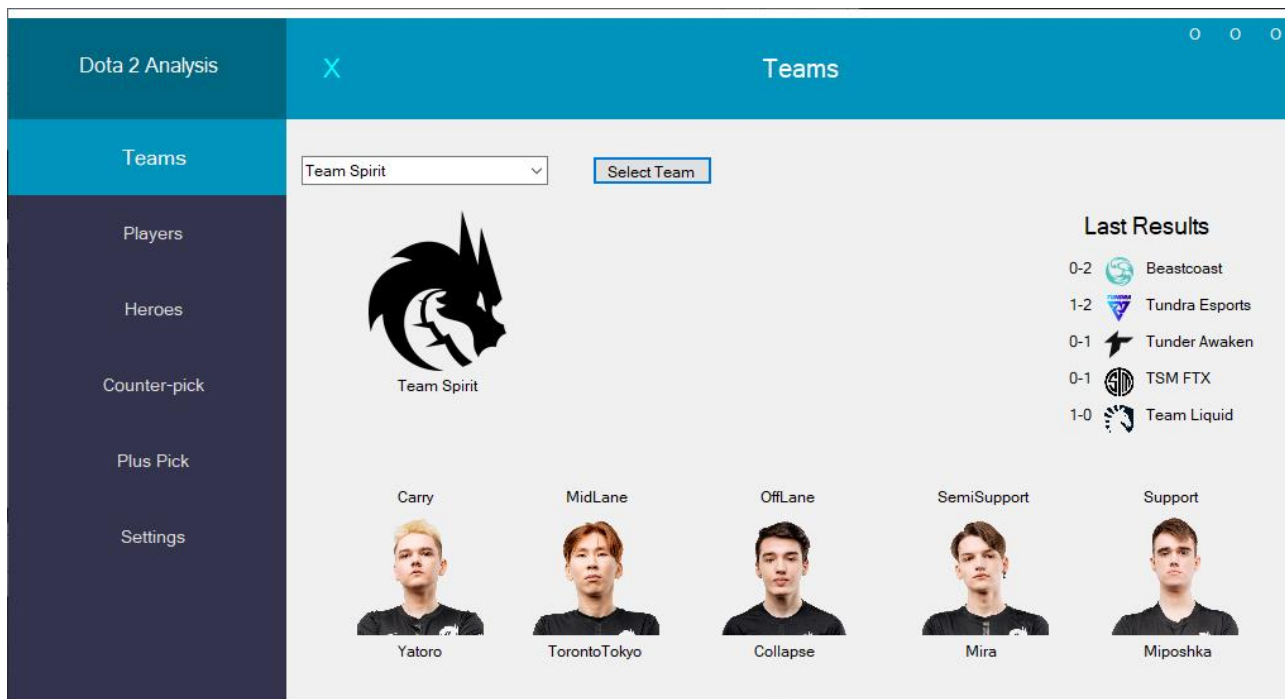


Рисунок 58 – Информация о команде Team Spirit

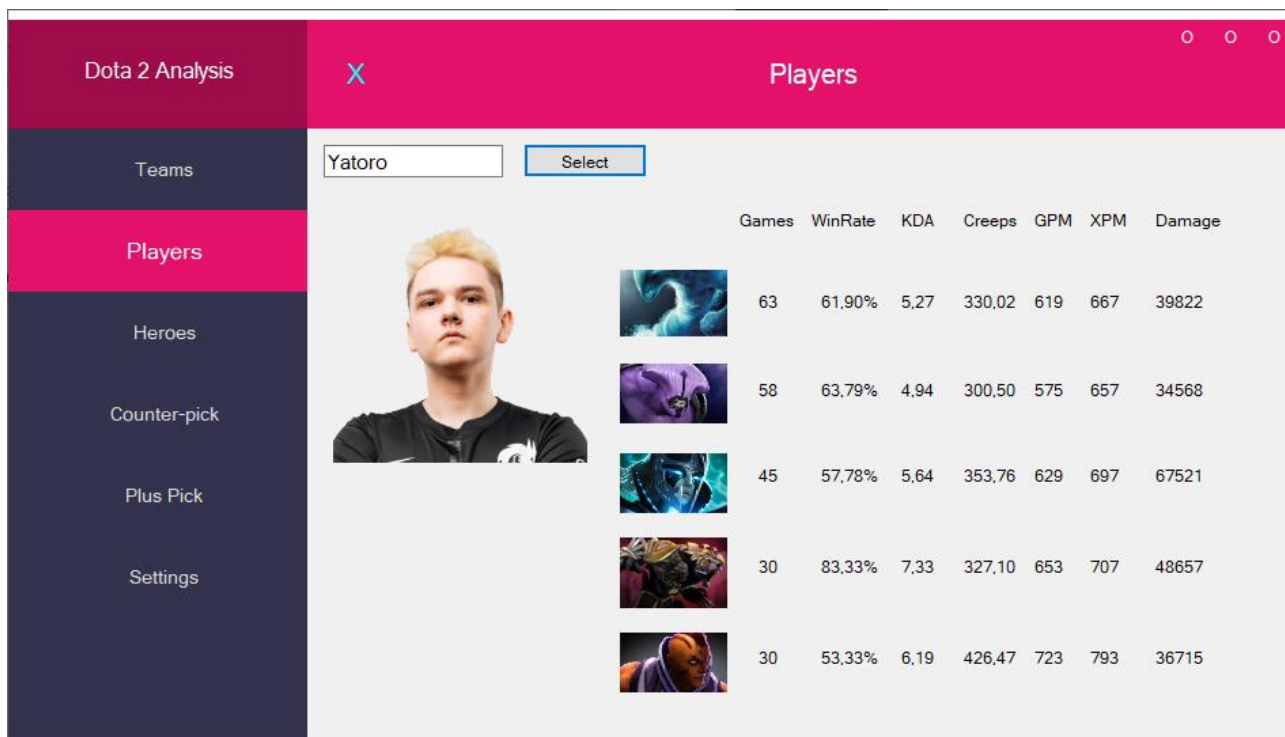


Рисунок 59 – Информация об игроке Yatoro



Тестирование системы предложения контрпики протестирована на герое Phantom Assassin (рис. 61). Получен список из пяти героев (Huskar, Anti Mage, Lone Druid, Phantom Lancer, Sniper), а также количество проведенных матчей этих героев с указанием показателя побед против героя Phantom Assassin.

The screenshot shows the 'Heroes' page for Phantom Assassin in the Dota 2 Analysis application. The left sidebar contains navigation options: Teams, Players, Heroes (selected), Counter-pick, Plus Pick, and Settings. The main content area features a dropdown menu with 'Phantom Assassin' selected and a 'Select' button. Below this is a large portrait of the hero. To the right, there is a table titled 'Most Used Items' with columns for Price, Matches, Wins, and WinRate. Below the portrait, 'Average stats' are listed.

Average stats	
PickRate	18,22%
WinRate	49,97%
KDA	3,08
GPM	535
XPM	638

Most Used Items				
	Price	Matches	Wins	WinRate
	4050	432455	240908	55,71%
	1400	430074	214222	49,81%
	3500	423724	237926	56,15%
	4100	423445	219680	51,88%
	1400	217867	133304	61,19%
	2875	186700	109412	58,60%

Рисунок 60 – Информация о персонаже Phantom Assassin

The screenshot shows the 'Counter-pick' page for Phantom Assassin in the Dota 2 Analysis application. The left sidebar contains navigation options: Teams, Players, Heroes, Counter-pick (selected), Plus Pick, and Settings. The main content area features a 'Select hero' dropdown with 'Phantom Assassin' selected. Below this is a large portrait of the hero. To the right, there is a table with columns for Games, WinRate, and KDA, listing suggested counter-picks.

Select hero		Games	WinRate	KDA
	Phantom Assassin	69	76,23%	6,93
	Phantom Assassin	44	65,42%	9,19
	Phantom Assassin	24	61,52%	5,67
	Phantom Assassin	79	60,02%	6,65
	Phantom Assassin	21	58,42%	8,32

Рисунок 61 – Предложенные программой контрпики персонажа Phantom Assassin

Для тестирования системы предложения дополнения пика (рис. 62) были выбраны три героя: Batrider, Phoenix, Faceless Void. Программа предложила трех героев (Axe, Earthshaker, Tidehunter) с указанием показателя побед в данных выборах.

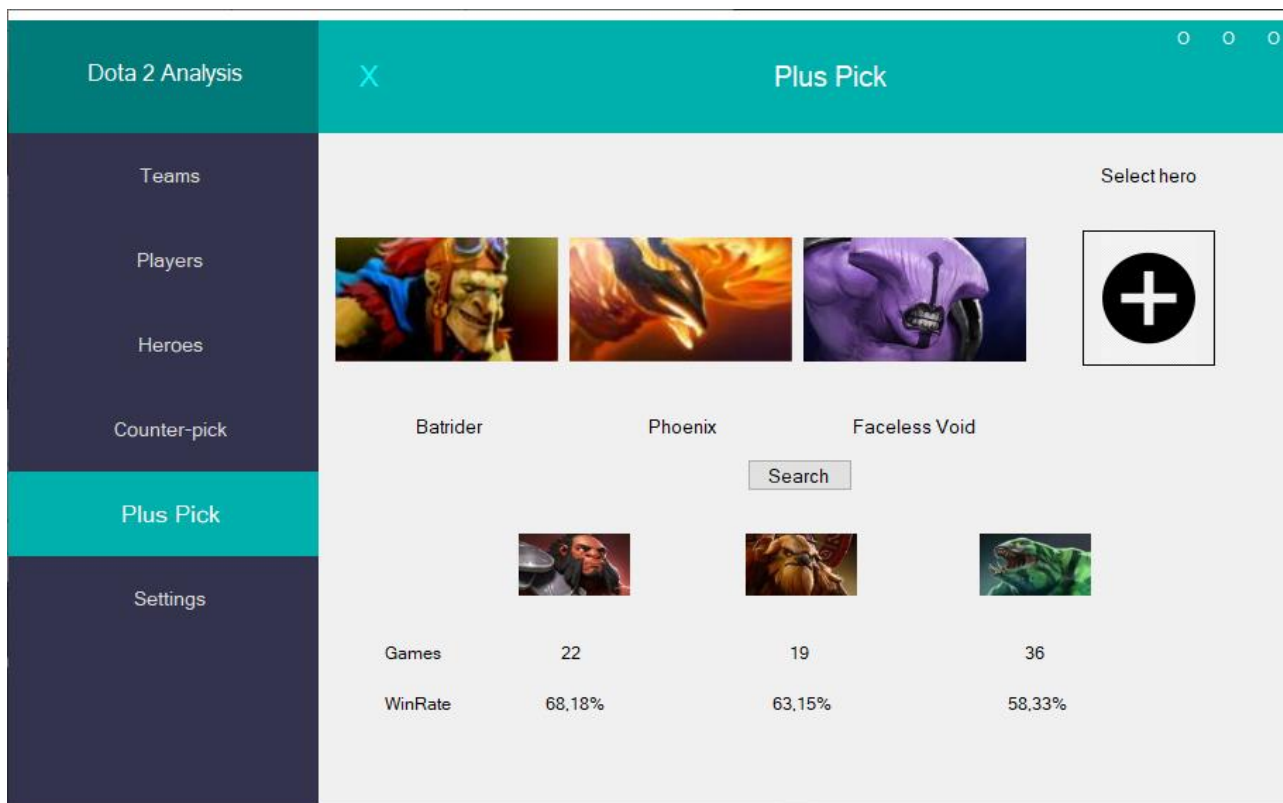


Рисунок 62 – Предложенные программой персонажи для дополнения пика

### 3.10 Анализ общей достоверности и практической значимости полученных решений

Практическая значимость подчеркивает прикладной характер полученных результатов, а также то, какие трансформации существующих процессов и явлений могут возникнуть в зонах их применимости. Для оценки практической ценности проведенного исследования важно проверить параметр достоверности полученных решений.

#### 3.10.1 Проверка достоверности полученных решений

Достоверность – результаты, которые получены в ходе исследования, они могут быть воспроизведены.

Проверим достоверность основных функций программы, сравнив их с действительными значениями.

Начнем с информации о команде, на основании главного информационного портала liquipedia.net в дисциплине «DOTA 2» состав команды Team Spirit (рис. 63) соответствует с полученным программой результатом.




Active Squad			
ID	Name	Position	Join Date
 Yatoro	Illya Mulyarchuk	1	2020-12-19 <sup>[B1]</sup>
 TORONTOTOKYO	Alexander Khertek	2	2020-12-19 <sup>[B1]</sup>
 Collapse	Magomed Khalilov	3	2020-12-19 <sup>[B1]</sup>
 Mira	Miroslaw Kolpakov	4	2021-03-??
 Miposhka 	Yaroslav Naidenov	5	2020-12-19 <sup>[B1]</sup>

Рисунок 63 – Состав команды Team Spirit

Также проверим полученный программой профайл игрока Yatoro. В этот раз будем отсылаться на популярнейший статистический портал dotabuff.com. Полученный программой результаты совпадают с информацией статистического портала (рис. 64).


MOST PLAYED HEROES	
Hero	Matches
 Morphling	63
 Faceless Void	58
 Phantom Assassin	45
 Ursa	30
 Slark	30

Рисунок 64 – Статистика героев игрока Yatoro

На основании проведенных сравнений можно утверждать, что полученные программой результаты достоверны.

### 3.10.2 Практическая значимость полученных решений

Разработанная система и полученные решения имеют значительную практическую значимость в области киберспорта.

Вот несколько ключевых аспектов и их влияние на киберспортивную деятельность:

- улучшение принятия решений: система, основанная на анализе данных и использовании моделей машинного обучения, может значительно повысить эффективность и точность принятия решений во время игры, анализ данных о прошлых матчах, поведении игроков и стратегиях противников позволяет выявить тенденции, прогнозировать возможные ходы и принимать более обоснованные решения;

- оптимизация тренировочных процессов: система может использоваться для анализа данных о тренировках, таких как игровые сессии, статистика игроков и команд, их производительность и прогресс, это позволяет тренерам и аналитикам выявлять сильные и слабые стороны игроков, определять области для улучшения и разрабатывать более эффективные тренировочные программы;

- оптимизация стратегий и тактик: анализ данных позволяет идентифицировать успешные стратегии и тактики, которые применялись в прошлых матчах, система может предоставить информацию о том, какие стратегии наиболее успешны против определенных команд или игроков, что позволяет составлять более эффективные планы на будущие матчи;

- улучшение командной работы: анализ данных о взаимодействии игроков в команде может помочь выявить проблемные моменты и недостатки в командной работе, система может предоставить информацию о вкладе каждого игрока, их ролях и обязанностях, а также предложить рекомендации по улучшению синхронизации и координации внутри команды;

- прогнозирование результатов матчей: система, основанная на анализе исторических данных и моделях машинного обучения, может помочь в прогнозировании результатов будущих матчей, это может быть полезным для беттинга, принятия решений о составе команды или выборе стратегии.

В целом, применение разработанной системы в киберспорте позволяет повысить качество тренировок, принятие решений и результаты команды. Это может стать конкурентным преимуществом для команды и помочь ей достичь

успеха в соревнованиях. Однако необходимо учитывать специфику киберспорта, такую как изменчивость мета-игры, индивидуальные особенности игроков и другие факторы, чтобы эффективно применять систему и получать максимальную выгоду.

## ЗАКЛЮЧЕНИЕ

В рамках магистерской диссертации была выполнена работа по разработке программы информационной поддержки киберспортивных команд. Основная цель работы заключалась в использовании алгоритма обработки MapReduce в создании программного продукта, который обеспечивает сбор данных о киберспортивных матчах, предоставляет аналитическую информацию и поддерживает профессиональные команды в процессе подготовки и принятия тактических решений.

В результате исследования и разработки была создана программа информационной поддержки, которая эффективно собирает данные с веб-ресурсов о прошедших киберспортивных матчах. Данные проходят обработку и фильтрацию с использованием алгоритма MapReduce, что позволяет отделить неактуальные данные и сохранить только релевантную информацию. После этого данные записываются в базу данных для последующего использования.

Одним из ключевых компонентов программы является модель машинного обучения, которая обучается на основе собранных данных. Эта модель предлагает рекомендации по контрпику и дополнению пика, что помогает командам принимать более обоснованные решения во время матчей. Пользовательский интерфейс программы предоставляет удобный доступ к информации о командах, игроках и героях, а также позволяет вводить пользовательские запросы и получать соответствующие рекомендации.

Выполненная работа в процессе написания диссертации подтверждает навыки и умения в области программной разработки и анализа данных. Разработка программы информационной поддержки для киберспортивных команд требовала глубокого понимания особенностей киберспортивной индустрии, аналитических методов и методов машинного обучения. В ходе работы проявлены способности анализировать и обрабатывать большие объемы данных, применять

современные алгоритмы и технологии, а также разрабатывать удобный пользовательский интерфейс.

Перспективы разработанной программы информационной поддержки огромны. Ее функциональность и возможности могут быть расширены путем добавления дополнительных обученных систем, улучшения алгоритмов обработки данных и аналитических методов. Кроме того, программу можно адаптировать для использования на других операционных системах, таких как Android OS, что расширит ее охват и сделает ее доступной для большего числа пользователей.

Таким образом, проделанная работа по разработке программы информационной поддержки киберспортивных команд имеет большую значимость для дальнейшего развития киберспорта. Программа обладает потенциалом для применения в реальных условиях и может способствовать улучшению результатов команд и повышению интереса к киберспорту.

## БИБЛИОГРАФИЧЕСКИЙ СПИСОК

- 1 8 эффективных способов измерения пользовательского опыта [Электронный ресурс]. – Режим доступа: <http://lpgenerator.ru/blog/2015/10/02/8-effektivnyh-sposobov-izmereniya-polzovatels>. – 17.04.2022.
- 2 Абденев, А. Ж. Оценивание риска в информационных системах на основе объективных и экспертных оценок / А. Ж. Абденев, Р. Н. Заркумова-Райхель // Вопросы защиты информации. – 2015. – 31(108). – С. 64-70.
- 3 Айсберг Big Data: почему мы видим лишь вершину. [Электронный ресурс]. – Режим доступа: <http://sap-technology.rbc.ru/big-data.html>. – 10.07.2021.
- 4 Андреева, А. С. К вопросу о понятии архитектуры систем / А. С. Андреева // Автоматизация, мехатроника, информационные технологии: материалы V Международной научно-технической интернет-конференции молодых ученых. – 2015. – С. 154-158.
- 5 Бессмертный, И. А. Системы искусственного интеллекта : учеб. пособие для СПО / И. А. Бессмертный. – 2-е изд., испр. и доп. – М. : Издательство Юрайт, 2018. – 130 с.
- 6 Бурков, В. Н. Прикладное машинное обучение: основные алгоритмы и приемы работы / В. Н. Бурков. – Москва: ДМК Пресс, 2020. – 352 с.
- 7 Бурунов, А. Р. Машинное обучение и анализ данных / А. Р. Бурунов. – Москва: ДМК Пресс, 2019. – 384 с.
- 8 Введение в юзабилити [Электронный ресурс]. – Режим доступа: <http://lpgenerator.ru/blog/2014/06/23/vvedenie-v-yuzabiliti> – 28.01.2022. – 24.02.2022.
- 9 Власов, Д. В. Методы противодействия анализу исполняемых файлов в информационных системах / Д. В. Власов, А. С. Минаев // Информация и безопасность. – 2014. – Т.17. – №32. – С. 308-311.
- 10 Воронцов, К. В. Математические методы обучения по прецедентам / К. В. Воронцов. – Москва: Издательство МГУ, 2014. – 208 с.



- 11 Гладкий, В. С. Парсинг веб-страниц: учебное пособие / В. С. Гладкий. – Москва: БИНОМ. Лаборатория знаний, 2017. – 224 с.
- 12 Головач, В. В. Дизайн пользовательского интерфейса: Искусство мыть слона [Электронный ресурс] – Режим доступа: <http://uibook2.usetheics.ru/2010>. – 10.10.2021.
- 13 Грушко, А. А. Безопасные архитектуры распределенных систем / А. А. Грушко // Системы и средства информатики. – 2014. – Т.24, №3. – С. 18-31.
- 14 Дивакар, М. Архитектура и шаблоны больших данных. [Электронный ресурс]. / М. Дивакар, К. Шрикант, Д. Швета. – Режим доступа: <https://www.ibm.com/developerworks/ru/library/bdarchpatterns1/index.html>. – 10.07.2021.
- 15 Добрякова, Г. Э. Проблемы идентификации в информационных системах / Г. Э. Добрякова // Правовая информатика. – 2014. – №3. – С. 28-32.
- 16 Емцева, Е. Д. Моделирование и анализ бизнес-процессов/ Е. Д. Емцева. – Владивосток: Изд-во ВГУЭС, 2012. – 76 с
- 17 Жарков, А. П. Анализ данных в Python: руководство / А. П. Жарков. – Москва: ДМК Пресс, 2021. – 432 с.
- 18 Жижимов, О. Л. Основные принципы, архитектура и реализация информационных систем СО РАН / О. Л. Жижимов, А. М. Федотов // Известия Кыргызского государственного технического ун-та им. И. Раззакова. – 2016. – №31. – С. 348-352.
- 19 Закон Фиттса или как его использовать [Электронный ресурс]. – Режим доступа: <https://habrahabr.ru/post/247257/>. – 25.04.2022.
- 20 Златопольский, Д. М. Основы программирования на языке Python. – М.: ДМК Пресс, 2017. – 284 с.
- 21 Зыков, С. В. Программирование. Объектно-ориентированный подход : учебник и практикум для академического бакалавриата / С. В. Зыков. – М. : Издательство Юрайт, 2019. – 155 с.
- 22 Иванов, С. М. Машинное обучение: основы и методы / С. М. Иванов. – Москва: Лань, 2017. – 352 с.

- 23 Исаков М. Б. Интеллектуальный интерфейс в информационных системах / М. Б. Исаков // Труды Университета. – 2014. – №4(57). – С. 74-77.
- 24 Использование файлов robots.txt [Электронный ресурс]. – Режим доступа: <https://habr.com/ru/articles/341774/>. – 4.12.2021.
- 25 Калашян, А. Н. Структурные модели бизнеса: DFD-технологии / А. Н. Калашян, Г. Н. Калянов. – Москва, 2009. – 69 с.
- 26 Карпов, А. Ю. Машинное обучение: алгоритмы и модели / А. Ю. Карпов. – Санкт-Петербург: БХВ-Петербург, 2019. – 432 с.
- 27 Карпычев, В. Ю. Управление совокупной стоимостью владения информационной системой: современное состояние и перспективы / В. Ю. Карпычев // Экономический анализ: теория и практика. – 2015. – №8(407). – С. 25-37.
- 28 Козлов А. М. Tress. [Электронный ресурс]. – Режим доступа: <https://github.com/astur/tress>. – 13.12.2021.
- 29 Козлов А. М. Web scraping при помощи Node.js [Электронный ресурс]. – Режим доступа: <https://habrahabr.ru/post/301426/>. – 25.11.2021.
- 30 Лаврищева, Е. М. Программная инженерия и технологии программирования сложных систем : учебник для вузов / Е. М. Лаврищева. – 2-е изд., испр. и доп. – М. : Издательство Юрайт, 2019. – 432 с.
- 31 Майныч, В. Г. UML в информационных системах / В. Г. Майныч // Поколение будущего: взгляд молодых ученых – 2014: сб. науч. ст. 3-й Международной молодежной научной конференции. – 2014. – С. 48-52.
- 32 Максимов, А. А. Разработка информационной системы с гибридной клиент-серверной архитектурой на основе одноранговых сетей с динамической топологией / А. А. Максимов, С. С. Зобнин // Информатика и кибернетика (ComCom-2015): сб. докл. студенческой научной конференции Института информационных технологий и управления. – 2015. – С. 284-287.
- 33 Малявко, А. А. Формальные языки и компиляторы : учеб. пособие для вузов / А. А. Малявко. – М. : Издательство Юрайт, 2018. – 429 с.

- 34 Массель, Г. Г. Психологические аспекты пользовательского интерфейса современных компьютерных систем / под ред. Л. В. Массель. Иркутск : ИСЭМ СО РАН, 2000. – 52 с.
- 35 Моделирование данных [Электронный ресурс]. – Режим доступа: [http://citforum.ru/database/case/glava2\\_4\\_1.shtml](http://citforum.ru/database/case/glava2_4_1.shtml). – 10.07.2021.
- 36 Монахов, М. Ю. Модель управления процессом обеспечения достоверности информационных ресурсов в информационно-телекоммуникационных системах / М. Ю. Монахов, Ю. М. Монахов, И. И. Семенова // Проектирование и технология электронных средств. – 2014. – №3. – С. 34-40.
- 37 Мухин, С. В. Обучение на размеченных данных: глубокое обучение / С. В. Мухин. – Москва: Издательство Юрайт, 2018. – 432 с.
- 38 Николаева И. К., Ландо Т. А. Прикладная и компьютерная лингвистика. Изд. 2-е. – М.: ЛЕНАНД, 2017. – 320 с.
- 39 Парсинг. Что это? [Электронный ресурс]. – Режим доступа: <https://www.ipipe.ru/info/parsing.html>. – 1.12.2021.
- 40 Применение robots.txt [Электронный ресурс]. – Режим доступа: <https://yandex.ru/methods/use/robots-txt.xml>. – 6.12.2021.
- 41 Реляционные базы данных [Электронный ресурс]. – Режим доступа: <https://yandex.ru/support/database>. – 16.12.2021.
- 42 Ронжин, А. Л. Многомодальные интерфейсы: основные принципы и когнитивные аспекты / А. Л. Ронжин, А. А. Карпов // Труды СПИИРАН. Вып. 3. Т. 1. СПб.: Наука, 2006. – 24 с.
- 43 Салова, В. В. Интеллектуальная система поддержки принятия решений по проведению аудита информационных систем персональных данных / В. В. Салов, В. И. Васильев // Вестник Уфимского государственного авиационного технического ун-та. – 2014. – №3(64). – С. 261-269.
- 44 Сергеев, Р. К. Разработка программы информационной поддержки киберспортивных команд / Р. К. Сергеев // Молодежь XXI века: шаг в будущее : материалы конф. – Благовещенск: Изд-во Дальневосточный ГАУ, 2022. - Т. 4. - С. 216-217.

- 45 Сергеев, Р. К. Целесообразность разработки программы информационной поддержки киберспортивных команд / Р. К. Сергеев // Молодежь XXI века: шаг в будущее : материалы конф. – Благовещенск: Изд-во БГПУ, 2021. - С. 786-787.
- 46 Сергеев, С. Ф. Методы тестирования и оптимизации интерфейсов информационных систем. СПб.: НИУ ИТМО, 2013. – 120 с.
- 47 Силич, В. А. Моделирование и анализ бизнес-процессов: учебное пособие / В. А. Силич, М. П. Силич. – Томск: Томский государственный университет систем управления и радиоэлектроники, 2011. – 128 с.
- 48 Советов, Б. Я. Базы данных : учебник для прикладного бакалавриата / Б. Я. Советов, В. В. Цехановский, В. Д. Чертовской. – 3-е изд., перераб. и доп. – М. : Издательство Юрайт, 2019. – 420 с.
- 49 Стрижов, В. В. Введение в машинное обучение / В. В. Стрижов. – Москва: ФИЗМАТЛИТ, 2019. – 416 с.
- 50 Сухенко, А. К. Понятие профиля информационной системы, структура профилей информационных систем / А. К. Сухенко // Информационно-телекоммуникационные системы и технологии: материалы Всероссийской научно-практической конференции. - 2014. - С. 177-178.
- 51 Теплицкий, И. А. Алгоритмы машинного обучения: эволюция и новейшие тенденции / И. А. Теплицкий. – Санкт-Петербург: Питер, 2020. – 352 с.
- 52 Филлипс Ф. Программирование компьютерного зрения на языке Python / Ф. Филлипс. – Санкт-Петербург : Питер, 2018. – 384 с.
- 53 Федоров, Д. Ю. Программирование на языке высокого уровня python : учеб. пособие для прикладного бакалавриата / Д. Ю. Федоров. – 2-е изд., перераб. и доп. – М. : Издательство Юрайт, 2019. – 161 с.
- 54 Цуканова, О. А. Методология и инструментарий моделирования бизнес-процессов: учеб. пособие. – Санкт-Петербург: ИТМО, 2015. – 37 с.
- 55 Черткова, Е. А. Статистика. Автоматизация обработки информации : учеб. пособие для вузов / Е. А. Черткова : под общ. ред. Е. А. Чертковой. – 2-е изд., испр. и доп. – М. : Издательство Юрайт, 2017. – 195 с.

- 56 Что такое Big Data - большие данные [Электронный ресурс]. – Режим доступа: <https://www.oracle.com/ru/big-data/what-is-big-data.html>. – 10.07.2021.
- 57 Что такое Scikit Learn - гайд по популярной библиотеке Python для начинающих [Электронный ресурс]. – Режим доступа: <https://datastart.ru/blog/read/chto-takoe-scikit-learn-gayd-po-populyarnoy-biblioteke-python-dlya-nachinayuschih>. – 12.02.2023
- 58 Шалимова, А. И. Медиаиндустрия в 2019-2023 гг./ А. И. Шалимова, О. С. Малышев, Г. Д. Сидоров // Обзор мировой и российской индустрии. – 2019. – № 4. – С. 21-29.
- 59 Эмоциональный веб-дизайн [Электронный ресурс]. – Режим доступа: [http://royallib.com/read/uolter\\_aaron/emotsionalniy\\_veb\\_dizayn.html#0](http://royallib.com/read/uolter_aaron/emotsionalniy_veb_dizayn.html#0). – 22.02.2022.
- 60 Big Data. [Электронный ресурс]. – Режим доступа: <https://www.calltouch.ru/glossary/big-data>. – 10.07.2021.
- 61 Crawling и scraping. [Электронный ресурс]. – Режим доступа: <https://ru.code-maven.com/adventures-in-crawling-and-scraping-the-world>. – 29.11.2021.
- 62 Data Mining: учебное пособие. [Электронный ресурс]. – Режим доступа: <https://habr.com/post/348028/>. – 3.12.2021.
- 63 Flask [Электронный ресурс]. – Режим доступа: <https://flask.palletsprojects.com/en/2.1.x/foreword/>. – 19.12.2021.
- 64 Hilbert, M. Big Data for Development: A Review of Promises and Challenges. / M. Hilbert – New York : Shelter Island, 2015. – 114 с.
- 65 Marz, N. Big Data for Development: A Review of Promises and Challenges / J. Warren, N. Marz. – University of California : Development Policy Review, 2016. – 139 с.
- 66 Newzoo Global Esports Market Report [Электронный ресурс]. – Режим доступа: <https://newzoo.com/insights/trend-reports/newzoo-global-esports-market-report-2020-light-version/>. – 10.07.2021.

67 Plimpton, S. J. MapReduce in MPI for Large-scale graph algorithms / S. J. Plimpton, K. D. Devine // Parallel Computing. – 2011. – Vol. 37, №9 – С. 610-632.

68 Python библиотеки для Data Science [Электронный ресурс]. – Режим доступа: <https://tproger.ru/translations/top-10-python-bibliotek-dlja-datascience/>. – 19.12.2021.

69 Scrapy 1.5 documentation [Электронный ресурс]. – Режим доступа: <https://doc.scrapy.org/en/latest/>. – 8.12.2021.

## ПРИЛОЖЕНИЕ А

### ПРОГРАММНЫЙ КОД

```
import scrapy
import mysql.connector

class HeroesSpider(scrapy.Spider):
    name = "heroes"
    start_urls = ["https://www.opendota.com/heroes"]

    def parse(self, response):
        # Получаем список героев
        heroes_list = response.xpath('//div[@class="hero-grid"]//a')

        for hero in heroes_list:
            # Получаем название героя
            hero_name = hero.xpath('div[@class="name"]/text()').get()

            # Получаем список способностей героя
            hero_abilities = hero.xpath('div[@class="abilities"]//a').getall()

            # Формируем строку запроса
            insert_query = f"INSERT INTO heroes (name, abilities) VALUES ('{hero_name}', '{',''.join(hero_abilities)}')"

            # Подключаемся к базе данных
            mydb = mysql.connector.connect(
                host="localhost",
                user="dota",
                password="dota",
                database="dota"
            )
            # Выполняем запрос на добавление данных в таблицу
            cursor = mydb.cursor()
            cursor.execute(insert_query)
            mydb.commit()
            yield {
                'name': hero_name,
                'abilities': hero_abilities,
            }
```

Рисунок А.1 – Программный код парсера информации о героях

## Продолжение приложения А

```
import scrapy
class DotaMatchSpider(scrapy.Spider):
    name = "dota_match"
    def start_requests(self):
        urls = [
            'https://www.opendota.com/matches/1',
        ]
        for url in urls:
            yield scrapy.Request(url=url, callback=self.parse)
    def parse(self, response):
        # Получаем название команд
        team_names = response.css('div.header-content-left span.team-
text::text').getall()
        # Получаем список игроков и их никнеймы
        players = response.css('div.players tbody tr')
        player_names = [p.css('a.link-type-player::text').get() for p in
players]
        player_nicknames = [p.css('a.link-type-player span.player-
name::text').get() for p in players]
        # Получаем данные об убийствах, смертях, помощи, добытых крипах,
ГПМ, ХПМ, уроне по героям,
        # уроне по строениям и лечению для каждого игрока
        player_stats = []
        for p in players:
            kills = p.css('td:nth-child(3)::text').get()
            deaths = p.css('td:nth-child(4)::text').get()
            assists = p.css('td:nth-child(5)::text').get()
            last_hits = p.css('td:nth-child(6)::text').get()
            gpm = p.css('td:nth-child(7)::text').get()
            xpm = p.css('td:nth-child(8)::text').get()
            hero_damage = p.css('td:nth-child(9)::text').get()
            tower_damage = p.css('td:nth-child(10)::text').get()
            hero_healing = p.css('td:nth-child(11)::text').get()
            items = p.css('td.items img::attr(title)').getall()
            player_stats.append({
                'kills': kills,
                'deaths': deaths,
                'assists': assists,
                'last_hits': last_hits,
                'gpm': gpm,
                'xpm': xpm,
                'hero_damage': hero_damage,
                'tower_damage': tower_damage,
                'hero_healing': hero_healing,
                'items': items
            })
        next_match_url = response.xpath('//a[contains(text(),"Next
Match")]/@href').get()
        if next_match_url:
            # Добавляем URL следующего матча в очередь на обработку
            next_match_url = response.urljoin(next_match_url)
            yield scrapy.Request(next_match_url, callback=self.parse)
```

Рисунок А.2 – Программный код парсера информации о матче



## Продолжение приложения А

```
import json

def mapper(match):
    team1 = match['team1']
    team2 = match['team2']
    players1 = match['players1']
    players2 = match['players2']

    yield team1, players1
    yield team2, players2
def reducer(team, players):
    last_match = get_last_match(team)
    active_players = get_active_players(last_match, players)
    yield team, active_players

with open('active_players.json', 'w') as f:
    json.dump(active_players, f, indent=4)
```

Рисунок А.3 – Код работы алгоритма MapReduce

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=42)
```

Рисунок А.4 – Код для разделения данных на выборки

```
import xgboost as xgb

# Разделение данных на обучающую и тестовую выборки
train_X, test_X, train_y, test_y = train_test_split(X, y, test_size=0.3)

# Создание объекта модели градиентного бустинга
model = xgb.XGBClassifier()

# Обучение модели
model.fit(train_X, train_y)

# Получение предсказаний на тестовой выборке
predictions = model.predict(test_X)

# Оценка качества модели
accuracy = accuracy_score(test_y, predictions)
print(«Accuracy: %.2f%%» % (accuracy * 100.0))
```

Рисунок А.5 – Код обучения модели машинного обучения

## Продолжение приложения А

```
Def recommend_heroes(user_hero, df, model, num_recommendations=3):
    # создаем DataFrame, содержащий информацию только о матчах с вы-
    бранным героем
    df_hero = df[df.hero == user_hero]
    # создаем список всех героев, за которых играли противники выбран-
    ного героя
    enemy_heroes = df_hero.enemy_heroes.unique().tolist()
    # создаем DataFrame, содержащий информацию о матчах, в которых
    участвовали выбранные герои противника
    df_enemies = df[df.enemy_heroes.isin(enemy_heroes)]
    # создаем матрицу признаков и вектор целей
    X = df_enemies.drop(['match_id', 'hero', 'team', 'enemy_heroes',
    'result'], axis=1)
    y = df_enemies['result']
    # производим предсказание вероятностей победы для всех героев,
    кроме выбранного
    preds = model.predict_proba(X)[: , 1]
    # создаем DataFrame, содержащий информацию о героях и их рейтинге
    hero_ratings = pd.DataFrame({'hero': df_enemies['hero'].unique(),
    'rating': preds})
    # сортируем героев по рейтингу в порядке убывания и выбираем n
    лучших
    top_heroes = hero_ratings.sort_values('rating', ascend-
    ing=False).head(num_recommendations)
    # выводим список лучших героев для выбора пользователем
    return top_heroes['hero'].tolist()
```

Рисунок А.6 – Реализация предложения контрпика

## Продолжения приложения А

```
Def recommend_plushero(user_heroes, df, model, num_recommendations=3):
    # создаем DataFrame, содержащий информацию только о матчах с выбран-
ными героями
    df_heroes = df[df.hero.isin(user_heroes)]
    # создаем список всех героев, за которых играли противники выбранных
героев
    enemy_heroes = df_heroes.enemy_heroes.unique().tolist()
    # создаем DataFrame, содержащий информацию о матчах, в которых участ-
вовали выбранные герои противника
    df_enemies = df[df.enemy_heroes.isin(enemy_heroes)]
    # создаем матрицу признаков и вектор целей
    X = df_enemies.drop(['match_id', 'hero', 'team', 'enemy_heroes',
'result'], axis=1)
    y = df_enemies['result']
    # производим предсказание вероятностей победы для всех героев, кроме
выбранных
    preds = model.predict_proba(X)[:, 1]
    # создаем DataFrame, содержащий информацию о героях и их рейтинге
    hero_ratings = pd.DataFrame({'hero': df_enemies['hero'].unique(),
'rating': preds})
    # сортируем героев по рейтингу в порядке убывания и выбираем лучшего
    best_hero = hero_ratings.sort_values('rating', ascending=
ing=False).head(1)['hero'].iloc[0]
    # выводим список лучших героев для выбора пользователем
    return best_hero
```

Рисунок А.7 – Реализация предложения дополнения пика