

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

«ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ  
УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ» (ТУСУР)

Кафедра автоматизации обработки информации (АОИ)

К ЗАЩИТЕ ДОПУСТИТЬ  
Заведующий кафедрой АОИ  
канд. экон. наук, доцент  
\_\_\_\_\_ А.А. Сидоров  
«\_\_\_\_\_» \_\_\_\_\_ 2022 г.

Магистерская работа по направлению 09.04.04  
«Программная инженерия»

## **ПРОГРАММНАЯ СИСТЕМА ДЛЯ ПРОВЕДЕНИЯ СОРЕВ- НОВАНИЙ ПО ВОЛЕЙБОЛУ**

Студент гр. 420-М1  
\_\_\_\_\_ А.М. Васильев  
«\_\_\_\_\_» \_\_\_\_\_ 2022 г.

Руководитель  
доцент кафедры АОИ, канд. техн. наук  
\_\_\_\_\_ Л.П. Турунтаев  
«\_\_\_\_\_» \_\_\_\_\_ 2022 г.

Томск 2022

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное  
учреждение высшего образования  
«ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ  
УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ» (ТУСУР)  
Кафедра автоматизации обработки информации (АОИ)

УТВЕРЖДАЮ  
Заведующий кафедрой АОИ  
канд. экон. наук, доцент  
\_\_\_\_\_ А.А. Сидоров  
«\_\_\_\_\_» \_\_\_\_\_ 2022 г.

ЗАДАНИЕ  
на выполнение магистерской работы

студенту Васильеву Андрею Михайловичу группы 420-M1,  
факультета систем управления (ФСУ)

1. Тема работы: «Программная система для проведения соревнований по волейболу»  
утверждена приказом по вузу от \_\_\_\_\_ № \_\_\_\_\_
2. Срок сдачи работы на кафедру: \_\_\_\_\_
3. Содержание работы (перечень подлежащих разработке вопросов):
  - 1) анализ предметной области;
  - 2) выбор инструментов разработки;
  - 3) проектирование серверной и клиентской части системы;
  - 4) разработка системы.
4. Дата выдачи задания: \_\_\_\_\_

Руководитель:  
доцент кафедры АОИ, канд. техн. наук \_\_\_\_\_ Л.П. Турунтаев

«\_\_\_\_\_» \_\_\_\_\_ 2022 г.

Задание принял к исполнению:  
студент гр. 420-M1 А.М. Васильев \_\_\_\_\_

«\_\_\_\_\_» \_\_\_\_\_ 2022 г.

## РЕФЕРАТ

Выпускная квалификационная работа содержит, 64 страниц, 52 рисунков, 1 таблиц, 14 источников.

ВОЛЕЙБОЛ, АВТОМАТИЗАЦИЯ, МОБИЛЬНОЕ ПРИЛОЖЕНИЕ,  
КЛИЕНТ\_СЕРВЕРНАЯ АРХИТЕКТУРА

Объектом разработки является программная система, предназначенная для автоматизации процесса проведения соревнований по волейболу.

Цель работы – проектирование и разработка программной системы, включающей в себя клиентское мобильное приложение и веб-сервер.

Выпускная квалификационная работа содержит:

- анализ предметной области;
- выбор инструментов разработки;
- проектирование;
- разработку.

Результатом работы стала программная система, состоящая из мобильного приложения и веб сервера. Данная программная система позволяет перенести большую часть документооборота, связанного с подготовкой и проведением соревнования в цифровую среду, а также, благодаря применению веб-технологий, позволяет ретранслировать статистические данные, получаемые в ходе соревнований на веб-страницу, выполняющую роль спортивного табло.

Выпускная квалификационная работа выполнена в текстовом редакторе Microsoft Word 2010 с применением MS PowerPoint.

Средства реализации мобильного приложения – Среда разработки Android Studio, язык программирования Java, расширяемый язык разметки XML, библиотека Apache POI.

Средства реализации веб-сервера – гипертекстовый язык разметки HTML, язык программирования PHP, технологии CSS, система управления базами данных MySQL.

## ABSTRACT

The final qualifying work contains 64 pages, 52 figures, 1 tables, 14 sources.

### VOLLEYBALL, AUTOMATION, MOBILE APP, CLIENT\_SERVER ARCHITECTURE

The object of development is a software system designed to automate the process of conducting volleyball competitions.

The purpose of the work is the design and development of a software system that includes a client mobile application and a web server that allows you to perform the procedures for preparing and holding volleyball competitions by a single mobile device operator.

The final qualifying work contains:

- analysis of the subject area;
- choice of development tools;
- design;
- development.

The result of the work was a software system consisting of a mobile application and a web server. This software system allows you to transfer most of the workflow associated with the preparation and conduct of the competition to the digital environment, and also, thanks to the use of web technologies, allows you to relay the statistical data obtained during the competition to a web page that acts as a sports scoreboard .

The descriptive part of the final qualifying work was done in the Microsoft Word 2010 text editor using MS PowerPoint.

Mobile application implementation tools - Android Studio development environment, Java programming language, extensible XML markup language, Apache POI library.

Web server implementation tools – Notepad++ development environment, HTML hypertext markup language, PHP programming language, CSS technologies, MySQL database management system.

Оглавление	
Введение .....	6
1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ .....	8
1.1 Требования к проекту .....	8
1.2 Обзор аналогов .....	9
1.3 Сравнение аналогов .....	15
2 ИНСТРУМЕНТЫ РАЗРАБОТКИ .....	17
2.1 Инструменты разработки для мобильных устройств .....	17
2.2 Инструменты разработки БД .....	18
2.3 Инструменты разработки веб-функционала .....	19
3 ПРОЕКТИРОВАНИЕ .....	21
3.1 Выбор архитектуры программной системы .....	21
3.2 Требования к компонентам технического обеспечения .....	25
3.2.1 Требования к мобильным устройствам .....	25
3.2.2 Требования к ПК .....	25
3.3 Моделирование логики системы .....	27
3.4 Макеты интерфейса .....	30
3.4.1 Пользовательский интерфейс мобильного приложения .....	30
3.4.2 Макет веб-табло .....	33
3.5 Создание логической модели БД .....	33
3.6 Создание физической структуры БД .....	35
3.7 Создание формы заявок и протоколов .....	37
4 Разработка программной системы .....	41
4.1 Создание интерфейса мобильного приложения .....	41
4.2 Взаимодействие с базой данных .....	42
4.3 Реализация функционала мобильного приложения .....	44
4.4 Разработка веб-табло .....	53
4.5 Тестирование системы .....	59
Заключение .....	62
Список литературы .....	63

## Введение

Многие учреждения стараются поддерживать спортивный образ жизни своих сотрудников, но нести большие затраты, помимо инвентаря, на организацию такого мероприятия готов далеко не каждый. В мире, где очень популярен спорт ежегодно проводится несчетное количество всевозможных соревнований и чемпионатов. Большинство из них проводится по старинке – с записью на бумагу или в Excel. Этапы организации и проведения соревнований связаны с формированием, заполнением и отслеживанием большого количества бумажных документов. Необходимость ручного выполнения таких этапов как обработка заявок на участие в соревновании, составление турнирной таблицы, составление расписаний матчей, формирование протоколов матчей, представляет собой проблему, которую можно устранить путем автоматизации процесса обработки документации.

Целью данной работы является проведение этапов проектирования и разработки программной системы, которая поможет автоматизировать документооборот на этапах подготовки и проведения соревнований местного уровня по волейболу.

Общей целью проекта является разработка программной системы, автоматизирующей процесс проведения соревнований по волейболу.

Для достижения цели поставленной в рамках данной работы необходимо выполнить следующие задачи:

- определить требования к проекту;
- изучить существующие аналоги;
- осуществить выбор инструментов разработки;
- провести этап проектирования программной системы;
- произвести разработку всех составляющих частей программной системы;

Объектом исследования является процесс автоматизации проведения соревнований по волейболу.

Предмет исследования — программные средства, предназначенные для сохранения и обработки статистических данных получаемых в ходе проведения спортивных соревнований.

Практическая значимость данной работы заключается в снижении времени и трудозатрат необходимых для организации и проведения спортивных соревнований, за счет автоматизации процессов документооборота. Полученные результаты могут быть использованы для организации спортивных мероприятий малого масштаба.

# 1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

## 1.1 Требования к проекту

Разрабатываемая программная система призвана автоматизировать процессы подготовки и проведения соревнований по волейболу. Такая система должна соответствовать некоторым функциональным требованиям.

Дальнейшие функциональные требования формируются на основе информации представленной в источнике[1].

Функционал программной системы для проведения соревнований можно разделить на три блока:

1. блок подготовки к проведению соревнования;
2. блок проведения соревнования;
3. блок подведения итогов соревнования.

Этап подготовки к проведению соревнования включает в себя автоматизацию документооборота. Обработка заявок на участие в соревновании должна быть перенесена в электронную среду, полученные данные должны заноситься в базу данных. На следующем шаге, на основе полученных данных, система должна составлять турнирную сетку по одному из способов распределения. После составления сетки, должно формироваться расписание матчей. Весь описанный функционал призван уменьшить нагрузку на персонал, ведь обычно все полученные в бумажном виде документы обрабатываются вручную.

На этапе проведения соревнований также внедряется элемент автоматизации. Ведение протокола соревнований переводится в электронный формат, а также переносится на мобильное устройство, благодаря чему обеспечивается повышение скорости обработки результатов. Так же данный функционал позволяет предоставить возможность зрителю наблюдать за ходом матча в реальном времени, ведь данные заносимые в протокол одновременно транслируются на веб-табло.



Этап подведения итогов соревнования также подвергается автоматизации. На этом этапе программная система предоставляет оператору доступ к данным собранным по ходу проведения соревнования, а именно, к краткой сводке о финальном счете противоборствующих команд и электронным файлам заполненных протоколов. Благодаря такому функционалу можно будет ускорить обработку результатов матчей, а также уменьшить вероятность ошибки в расчетах, возникающей из-за человеческого фактора.

### **1.2 Обзор аналогов**

Анализ интернет источников показал, что в рамках мобильной платформы отсутствуют решения, предоставляющие пользователю функционал, соответствующий описанным ранее требованиям. Таким образом материалом для формирования функциональных сравнительных критериев выступят решения, разработанные для персональных компьютеров.

**Спортивные таблицы** – программа для ведения турниров, проводимых по круговой системе[2]. Поддерживаются следующие виды спорта: футбол, футзал (мини-футбол), хоккей, баскетбол, хоккей с мячом (бенди), гандбол, водное поло, волейбол, шахматы, теннис, американский футбол и бейсбол. Программа имеет русский и английский интерфейс. При вводе новых результатов программа автоматически пересчитывает показатели команд и перестраивает турнирные таблицы.

#### **Основные возможности программы**

- Неограниченное количество турниров, организованных в виде «дерева турниров»;
- Виды спорта: футбол, футзал (мини-футбол), хоккей, баскетбол, американский футбол, бейсбол, хоккей с мячом (бенди), гандбол, водное поло, волейбол, теннис и шахматы;
- Турниры из 1 или нескольких групп, либо конференций и дивизионов. До 70 команд в турнире;
- Турниры в 1-8 кругов;

- Информация о матчах: полный счет, дата, тур, стадион, судьи, произвольные комментарии, техническое поражение;
- Возможность настройки количества очков, начисляемых за победу, ничью, поражение (в основное время и дополнительное время/овертайм);
- Возможность настройки критериев распределения команд по местам;
- Турнирные таблицы: шахматка, показатели команд (дома/в гостях/всего);
- Календарь игр / список матчей и результатов;
- Фильтрация матчей по номеру круга, номеру тура или дате, фильтрация календаря по группе/конференции/дивизиону/команде;
- Сортировка таблиц и календаря игр по любому показателю простым нажатием мыши на соответствующий столбец;
- Статистика по командам (дома/в гостях/всего): набранные очки всего/потерянные/в среднем, голы забитые/пропущенные/разница/в среднем, посещаемость всего/в среднем и др;
- Графики по командам (дома/в гостях/всего): накопление очков по турам/датам, гистограмма результатов, посещаемость;
- Настройка внешнего вида турнирных таблиц (раскраска цветами, шрифты) и размеров;
- Информация по судьям, стадионам;
- Импорт (загрузка) календаря и результатов из текстовых и CSV файлов;
- Экспорт в форматы HTML, CSV, TXT, BMP, GIF, JPEG, WMF, EMF, возможность копирования таблиц в буфер обмена;
- Удобная функция печати таблиц с предварительным просмотром.

Данная программа является слегка улучшенным табличным редактором. В ней не реализован функционал жеребьёвки команд, а также нет механизма обеспечения наглядности для зрителей.

**Tournament planner** предназначен для управления любым типом турниров и предоставляет вам практические и профессиональные возможности [3]. Используйте один щелчок мыши, чтобы запланировать матч, распечатать розыгрыш, показать различные события и многое другое.

**Основные возможности программы:**

- Простое планирование с учетом доступности игроков, наличия корта, других матчей и т.д.;
- Настройте свои даты и временные интервалы;
- Обзор запланированных, сыгранных и внеплановых матчей;
- Создавайте резервные копии для вашей безопасности или обмена данными;
- Положение по круговой системе рассчитывается в режиме реального времени после ввода результатов. Вы можете определить свои собственные правила расчета;
- Измените расписание матчей, не отменяя его;
- Запланируйте сразу все матчи для вашего кругового турнира.

В отличие от предыдущей программы в данной присутствует функция автоматической жеребьёвки, однако остальные функции также отсутствуют.

**Tourney Master 3** – это программа для проведения чемпионатов, скомпонованная по принципу Все-в-Одном. Она может быть использована для любых спортивных мероприятий с любым количеством участников и игровых локаций. С её помощью вы можете создавать сложные расписания, используя при этом проработанный дружественный интерфейс.

**Основные возможности программы:**

- Создание и распечатка турнирных сеток;

- Создание и распечатка отчетов;
- Сохранение документации в формате HTML;
- Возможность комбинирования турнирных систем;
- Возможность выбирать терминологию для требуемого вида спорта.

У программы отсутствует функционал отображения хода матчей, однако хорошо сделан функционал по автоматизации работы с документацией.

**Финиш** - программа, предназначенная проведения соревнований по легкой атлетике и другим циклическим видам спорта. Она ведет учет соревнований, их программы, заявок и показанных результатов, хранит базу данных спортсменов тренеров и судей, разрядных нормативов, формирует все необходимые протоколы и другие печатные документы и отчеты.

#### **Основные функции программы:**

- Ввод и хранение справочной информации: разрядных нормативов, виды спорта, виды соревнований, организации, города;
- Ввод всей необходимой для работы информации людях: спортсменах и их результатах, тренерах, судьях;
- Ввод информации о Соревнованиях, видах программы и их параметрах;
- Ввод заявок с использованием имеющейся базы данных спортсменов;
- Проведение жеребьевки по разным алгоритмам. Формирование забегов;
- Удобный ввод результатов в различных видах программы. При этом возможен учет нескольких попыток и работа с эстафетными командами;
- Автоматическое формирование итоговых протоколов. Расстановка занятых мест, присвоение выполненных разрядов и набранных очков;

- Автоматическая подготовка и печать всех необходимых документов, стартовых и итоговых протоколов, пропусков. Формирование данных для размещения в интернете;
- Внешний вид и содержание всех печатаемых документов настраивается. Пользователи программы могут самостоятельно добавлять новые разновидности печатных документов, протоколов;
- Хранение всех проведенных соревнований в общей базе данных. История показанных результатов по каждому спортсмену;
- Возможность проведения соревнований любого уровня в соответствии с правилами проведения соревнований;
- Имеется возможность экспорта всех данных в электронные таблицы (Excel, Open Office);
- Автоматическое создание резервных копий базы данных и документов;
- Простой и понятный, не напрягающий зрение интерфейс программы. Быстрота и удобство ввода данных;
- Подробная документация, техническая поддержка зарегистрированных пользователей;
- Возможность тестирования оценочной версии программы. Получение помощи по настройке программы под вашу систему работы;
- Техподдержка, возможность адаптации программы под ваши требования;
- Бесперебойная работа неограниченного числа рабочих мест в локальной сети. Возможно удаленное подключение через интернет.

В данной программе реализован широкий функционал по работе со спортивной документацией, однако она не адаптирована для ведения документации командных соревнований.

**Исток-Турнир** — компьютерная программа для жеребьёвки участников, составления турнирных таблиц и расписаний при проведении соревнований по индивидуальным видам спорта: игровым и единоборствам. Предназначена для главных судей, организаторов соревнований, спортивных федераций.

### **Основные возможности программы:**

- Настройка вида спорта: перечень подвидов (дисциплин, разделов), перечни возрастных и весовых категорий;
- База данных спортсменов;
- Заполнение списка заявок, результатов взвешивания, данных об уплате вступительного взноса;
- Создание групп в возрастных и весовых категориях, в одиночном и парном разрядах;
- Ручная или компьютерная жеребьевка участников;
- Составление турнирных таблиц по олимпийской, круговой или смешанной схемам, а также по системе выступлений;
- До трёх этапов в каждой группе: отборочный, основной (финальный) и утешительный;
- Автоматическая нумерация поединков и составление расписаний (графика боёв);
- Подсчёт разницы выигранных и проигранных сетов и очков (геймов) при круговой схеме.

В данной программе реализован широкий функционал по работе со спортивной документацией, однако она не адаптирована для ведения документации командных соревнований.

В результате проведенного исследования можно сделать следующие выводы. Программы для проведения соревнований довольно распространены. Однако в существующих на данный момент решениях процесс организации и проведения соревнований не в полной мере автоматизирован.

### 1.3 Сравнение аналогов

Для каждого отдельно взятого соревнования составляется «положение о проведении соревнований», оно уникально для каждой организации, однако существуют этапы подготовки и проведения соревнований, общие для всех. К общим, можно отнести такие этапы как[4]:

- сбор и обработка заявок на участие в соревновании;
- составление расписания матчей;
- сохранение промежуточных результатов матчей;
- обеспечение трансляции промежуточных результатов матчей;
- составление протоколов проводимого соревнования.

Примем описанные выше этапы как критерии оценки программного обеспечения и составим таблицу сравнения возможностей ПО на основе данных собранных в разделе 2. Результат сравнения возможностей ПО представлен в таблице 1.1.

Таблица 1.1 – Сравнение возможностей аналогов.

Программная система	Заявки	Расписание матчей	Промежуточные результаты	Трансляция матчей	Протоколы
Спортивные таблицы	-	+	+	-	+
Tournament planner	-	+	+	-	+
Tourney master 3	-	+	+	-	+
Исток турнир	+	+	+	-	+
Проектируемая программная система	+	+	+	+	+

Исходя из собранных данных делаем вывод, что аналоги не решают в полном объёме поставленные задачи. А так как функционал разрабатываемой системы планируется реализовать на мобильной платформе, на которой отсутствуют прямые аналоги, система станет актуальным решением для проведения соревнований по волейболу.



## 2 ИНСТРУМЕНТЫ РАЗРАБОТКИ

### 2.1 Инструменты разработки для мобильных устройств

Среда разработки Eclipse – бесплатный фреймворк для разработки модульных кроссплатформенных приложений. Изначально проект разрабатывался в IBM как корпоративный стандарт IDE для разработки на разных языках под платформы IBM. Позже проект был переименован в Eclipse и предоставлен для дальнейшего развития сообществу[5].

Благодаря активному развитию, а также поддержке со стороны компании и сторонних разработчиков, на данный момент у этой IDE имеются следующие преимущества:

- официальная русификация интерфейса и документации;
- высокие показатели производительности;
- возможность подключения модулей;
- возможность групповой разработки.

Среда разработки IntelliJ IDEA – это среда разработки предоставляющая возможность создавать программы на нескольких языках программирования. Если рассматривать программирование под Android между IntelliJ IDEA и Eclipse, то первый вариант предпочтительнее, т. к. у этой среды имеются неоспоримые преимущества относительно своего конкурента:

- Более быстрая отладка значений;
- Автозаполнение методов;
- Наличие рефакторинга.

Среда разработки Android Studio – IDE созданная компанией Google на основе IntelliJ IDEA. Сохранив все основные преимущества первоисточника Android Studio также приобрела свои отличительные особенности. В Android Studio появилась возможность создания разметки графического интерфейса приложения в интерактивном режиме, как при работе с проектами Windows Forms в Visual Studio. Помимо всего прочего Android Studio имеет встроенный

эмулятор мобильных устройств, позволяющий тестировать работу приложений прямо по ходу написания их логики.

Среди рассмотренных сред разработки была выбрана Android Studio, так как она предоставляет наиболее качественный функционал, а также является средой разработки, идеально подходящей для человека с небольшим опытом работы в сфере программирования под Android.

## **2.2 Инструменты разработки БД**

Выбор целевой СУБД – выбор СУБД подходящего типа. Если тип используемой СУБД еще не выбран, то наиболее подходящим для осуществления такого выбора является переходное положение между концептуальным и логическим этапами проектирования базы данных. Однако этот выбор можно осуществить и в любой другой момент до начала логического проектирования, при условии, что имеется вся необходимая информация о таких общих требованиях к системе, как производительность, простота реорганизации, уровень защищенности и ограничения целостности данных.

Простейший подход к выбору нужной СУБД предусматривает оценку того, насколько функциональные возможности, предоставляемые СУБД, удовлетворяют существующим требованиям.

Рассмотрим СУБД совместимую с мобильными устройствами под управлением ОС Android - SQLite.

SQLite — это изумительная библиотека, встраиваемая в приложение, которое её использует. Будучи файловой БД, она предоставляет отличный набор инструментов для более простой (в сравнении с серверными БД) обработки любых видов данных[6].

Когда приложение использует SQLite, их связь производится с помощью функциональных и прямых вызовов файлов, содержащих данные (например, баз данных SQLite), а не какого-то интерфейса, что повышает скорость и производительность операций.

### ***Преимущества***

- **Файловая:** вся база данных хранится в одном файле, что облегчает перемещение;
- **Стандартизированная:** SQLite использует SQL; некоторые функции опущены, однако, есть и некоторые новые;
- **Отлично подходит для разработки и даже тестирования:** во время этапа разработки большинству требуется масштабируемое решение. SQLite, со своим богатым набором функций, может предоставить более чем достаточный функционал.

### ***Недостатки***

- **Отсутствие пользовательского управления:** продвинутые БД предоставляют пользователям возможность управлять связями в таблицах в соответствии с привилегиями, но у SQLite такой функции нет;
- **Невозможность дополнительной настройки:** опять-таки, SQLite нельзя сделать более производительной, поковырявшись в настройках — так уж она устроена.

Данная СУБД была выбрана для разрабатываемого проекта по следующим причинам:

- Библиотека для работы с SQLite встроена в Android Studio, что избавляет разработчика от необходимости производить интеграцию сторонних библиотек;
- Возможности, предоставляемые SQLite, позволяют полностью реализовать функционал, описанный при формировании требований к программной системе.

## **2.3 Инструменты разработки веб-функционала**

В качестве среды в рамках которой возможно проводить разработку и тестирование выступает Open Server Panel. Open Server Panel – это портативная программная среда, специально созданная с учетом пожеланий и рекомендаций веб-разработчиков.

Этот программный комплекс предоставляет пользователю тщательно подобранный набор серверного программного обеспечения, а также удобную и продуманную управляющую утилиту, обладающую мощными возможностями по настройке и администрированию всех доступных компонентов[7].

Для разработки планируемого веб-функционала предполагается использование встроенной в Open Server Panel СУБД MySQL. MySQL – свободная реляционная система управления базами данных. MySQL чаще всего используют при разработке веб-решений, что объясняется тесной интеграцией с популярными языками программирования, высокими показателями скорости и, конечно, её бесплатностью [8].

#### Преимущества MySQL:

- Открытый исходный код;
- Простота. Данная СУБД легко устанавливается, имеет понятный интерфейс, а разнообразие плагинов и дополнительных приложений делает работу с БД проще;
- Функционал. Обладает обширным инструментарием, который может пригодиться при разработке любого проекта;
- Безопасность. Имеет встроенные системы безопасности, работающие по умолчанию;
- Масштабируемость. Возможна работа с различными объемами данных, как большими, так и малыми;
- Скорость. Является одной из самых быстрых среди имеющихся на современном рынке.

В связке с Open Server Panel при разработке серверного функционала применяется редактор кода Notepad++. Он позволяет быстро и удобно писать, и редактировать код на языках HTML и PHP, используемых при создании веб-страниц и программировании серверных взаимодействий.

## 3 ПРОЕКТИРОВАНИЕ

### 3.1 Выбор архитектуры программной системы

Описанный ранее функционал предполагает обмен данными между несколькими устройствами. Устройство клиент должно обмениваться данными с веб-сервером для дальнейшего их использования. Это в свою очередь означает, что разрабатываемое приложение должно проектироваться согласно «Клиент – Серверной» архитектуре.

Архитектура «Клиент - Сервер» предусматривает разделение процессов предоставления услуг и отправки запросов на них на различных устройствах в сети, каждое из которых выполняет свои задачи независимо от других [9]. Визуальная репрезентация структуры клиент – серверной архитектуры представлена на рисунке 3.1.

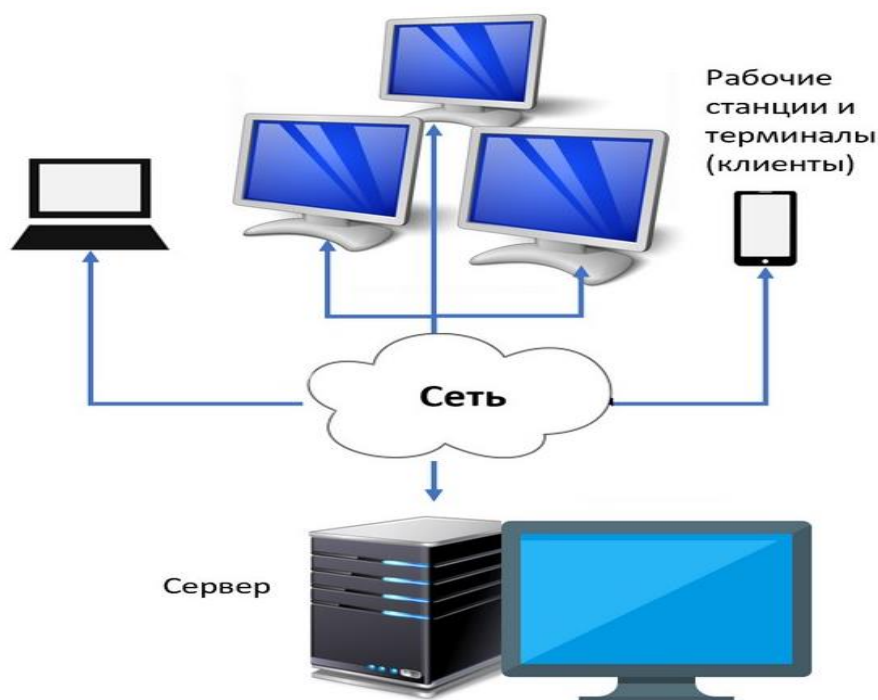


Рисунок 3.1 – Структура «Клиент – Серверной» архитектуры.

Архитектуру «клиент - сервер» принято разделять на три класса:

1. одноуровневая архитектура;
2. двухуровневая архитектура;
3. трёхуровневая архитектура.

Одноуровневая архитектура «клиент-сервер» (рисунок 3.2) – построена так, что все прикладные программы рассредоточены по рабочим станциям, которые обращаются к общему серверу баз данных или к общему файловому серверу. Никаких прикладных программ сервер при этом не исполняет, только предоставляет данные.

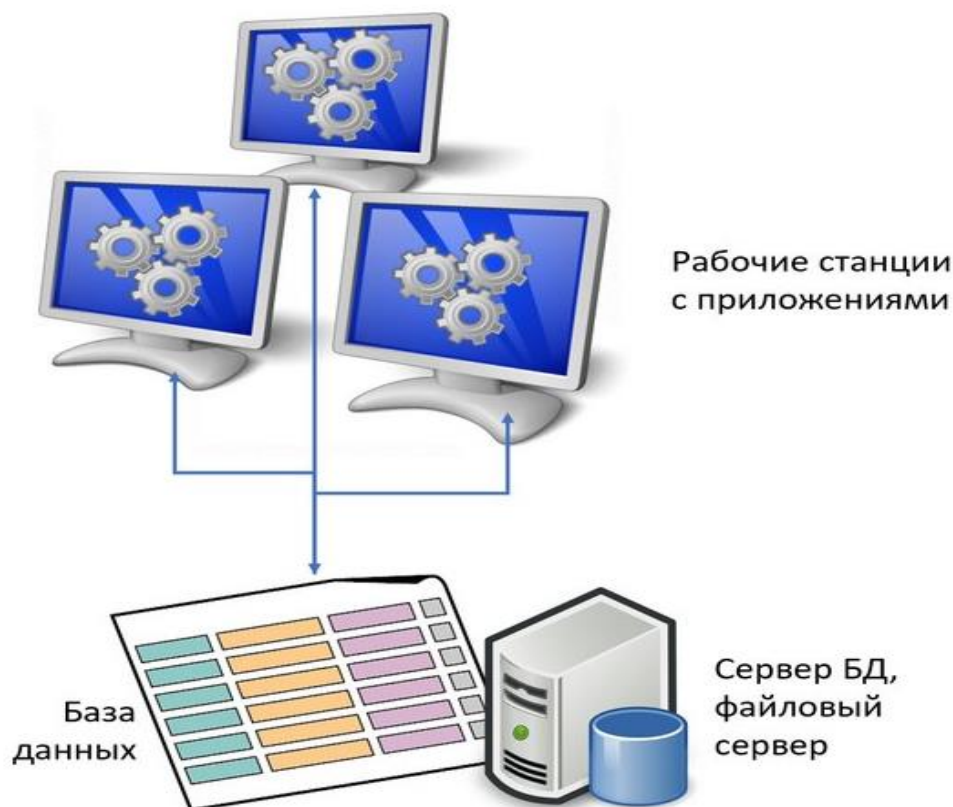


Рисунок 3.2 – Структура одноуровневой архитектуры «Клиент-Сервер».

К двухуровневой архитектуре «клиент-сервер» (рисунок 3.3) следует относить такую, в которой прикладные программы сосредоточены на сервере приложений, а в рабочих станциях находятся программы-клиенты, которые предоставляют для пользователей интерфейс для работы с приложениями на общем сервере.



Рисунок 3.3 – Структура двухуровневой архитектуры «Клиент-Сервер».

Такая архитектура представляется наиболее логичной для архитектуры «клиент-сервер». В ней, однако, можно выделить два варианта. Когда общие данные хранятся на сервере, а логика их обработки и бизнес-данные хранятся на клиентской машине, то такая архитектура носит название “fat client thin server” (толстый клиент, тонкий сервер). Когда не только данные, но и логика их обработки и бизнес-данные хранятся на сервере, то это называется “thin client fat server” (тонкий клиент, толстый сервер).

Преимущества двухуровневой архитектуры:

- Легко конфигурировать и модифицировать приложения;
- Пользователю обычно легко работать в такой среде;
- Хорошая производительность и масштабируемость.
- Однако, у двухуровневой архитектуры есть и ограничения:
- Производительность может падать при увеличении числа пользователей;
- Потенциальные проблемы с безопасностью, поскольку все данные и программы находятся на центральном сервере;
- Все клиенты зависимы от базы данных одного производителя;

В трёхуровневой архитектуре (рисунок 3.4) сервер баз данных, файловый сервер и другие представляют собой отдельный уровень, результаты работы которого использует сервер приложений. Логика данных и бизнес-логика находятся в сервере приложений. Все обращения клиентов к базе данных происходят через промежуточное программное обеспечение (middleware), которое находится на сервере приложений. Вследствие этого, повышается гибкость работы и производительность.

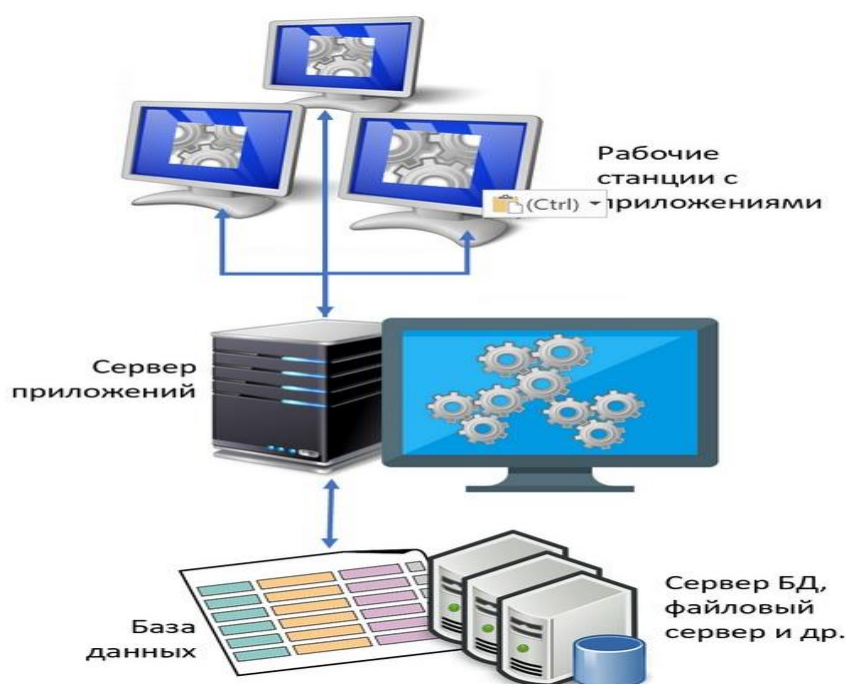


Рисунок 3.4 – Структура трехуровневой архитектуры «Клиент-Сервер».

Преимущества трёхуровневой архитектуры:

- Целостность данных;
- Более высокая безопасность, по сравнению с двухуровневой архитектурой;
- Защищённость базы данных от несанкционированного проникновения.

Предполагается, что веб-сервер осуществляет с приложением клиентом обмен данными, после чего транслирует полученные данные на веб-страницу. Такой функционал наиболее близок к одноуровневой «клиент-серверной» архитектуре. Таким разрабатываемая система соответствует одноуровневой «клиент-серверной» архитектуре.



## 3.2 Требования к компонентам технического обеспечения

### 3.2.1 Требования к мобильным устройствам

Мобильное устройство выступает основной платформой разрабатываемой системы, так как несущее основной функционал приложение, разрабатывается именно для этой платформы. Минимальная комплектация для мобильных устройств пользователей системы должна включать следующие компоненты:

- Центральный процессор: 2 ядра с тактовой частотой 1,5 ГГц;
- Графический процессор: PowerVR серии 8XE с тактовой частотой ядра от 570 МГц, поддерживающий программные интерфейсы OpenGL ES 3.2, Vulkan 1.1, OpenCL 1.2, Android NN HAL API
- Операционная система: Android v. 8.1, MIUI;
- Сенсорный дисплей: с соотношением сторон 18:9;
- Оперативная память: 2 Гб;
- Сети LTE Cat.7 1, 3, 4, 5, 7, 8, 20, 38, 40

### 3.2.2 Требования к ПК

При разработке системы предполагается возможность размещения веб-сервера на оборудовании пользователя. Для осуществления данной задачи пользователю потребуется наличие персонального компьютера, соответствующего минимальным требованиям к комплектации. Минимальная комплектация для персональных компьютеров пользователей системы должна включать следующие компоненты:

• **Системная плата:** поддержка частоты системной шины не менее 800 МГц; поддержка двухканальной DDR2 PC-4200 (DDR533), DDR2 PC-5300 (DDR667), памяти максимальным объемом не менее 2 Гб (не менее 4-х слотов); один слот PCI-E x 16, не менее 2 x PCI-E x 1; не менее 2-х слотов PCI

2.2; интерфейс IDE не ниже UltraATA/133; интерфейс Serial-ATA II; интегрированная звуковая подсистема; не менее 4-х портов USB 2.0, наличие COM и LPT портов и портов PS/2 для подключения клавиатуры и мыши;

•**Сетевая карта:** поддержание скорости передачи данных не менее 100 Мбит/сек, количество разъемов RJ-45 не менее одного, поддержка стандартов IEEE 802.3, IEEE 802.3x, IEEE 802.3u;

•**Оперативная память:** два модуля. DDR2 PC-5300 (DDR667) объемом не менее 512Mb (для активизации 2х канального режима работы памяти модули устанавливаются парами). Общий объем памяти 1 Gb;

•**Видеоадаптер:** объем видеопамати 64 Мб; интерфейс PCI-E; обеспечение разрешения монитора не менее 1024x768 точек при частоте вертикальной развертки не менее 85 Гц; шина памяти 128-бит; коннектор D-Sub, DVI-I (не допускается использование видеоадаптера, интегрированного в системную плату);

•**Подсистема дисковой памяти:** жесткий диск объемом не менее 120 Гб, средним временем доступа не более 9 мс, буфером не менее 8 Мб, скоростью вращения шпинделя не менее 7200 об/мин, интерфейсом Serial ATA II; один дисковод FDD 3,5" 1,44 Мб; дисковод Combo DVD+CD-RW;

•**Корпус с блоком питания:** блок питания ATX мощностью не менее 400 Вт; внутренних отсеков 3,5" – не менее 5, внешних отсеков 5,25" – не менее 4-х, внешних отсеков 3,5" – не менее 2-х, не менее 2-х портов USB на передней панели. Вентилятор 90x90 мм на задней стенке корпуса и место для вентилятора 80x80 мм на передней, воздуховод в боковой стенке над процессорным разъемом;

•**Клавиатура:** должна иметь не менее 104 клавиш с контрастным отображением букв русского алфавита и интерфейс PS/2. Мышь должна быть оптической, иметь не менее 2-х кнопок и 1-го колеса прокрутки, кабель длиной не менее 1,5 м и интерфейс USB или PS/2.

Требования к программному обеспечению персональных компьютеров пользователей:

- ОС: Microsoft Windows 10 и выше;
- Текстовый редактор: Microsoft Office Word 2013 и выше.

### 3.3 Моделирование логики системы

Для создания первичной модели будущей системы предполагается создание диаграммы по методологии IDEF0. IDEF0 – это методология функционального моделирования и графическая нотация, предназначенная для формализации и описания системных процессов. Отличительной особенностью IDEF0 является её акцент на соподчиненность объектов. В IDEF0 рассматриваются логические отношения между работами, а не их временная последовательность [10]. Контекстная диаграмма для проектируемой системы представлена на рисунке 3.2.



Рисунок 3.5 – Контекстная диаграмма процесса проведения соревнований.

Для описания модулей, входящих в состав системы необходимо выполнить декомпозицию контекстной диаграммы. Декомпозированная контекстная диаграмма представлена на рисунке 3.3.

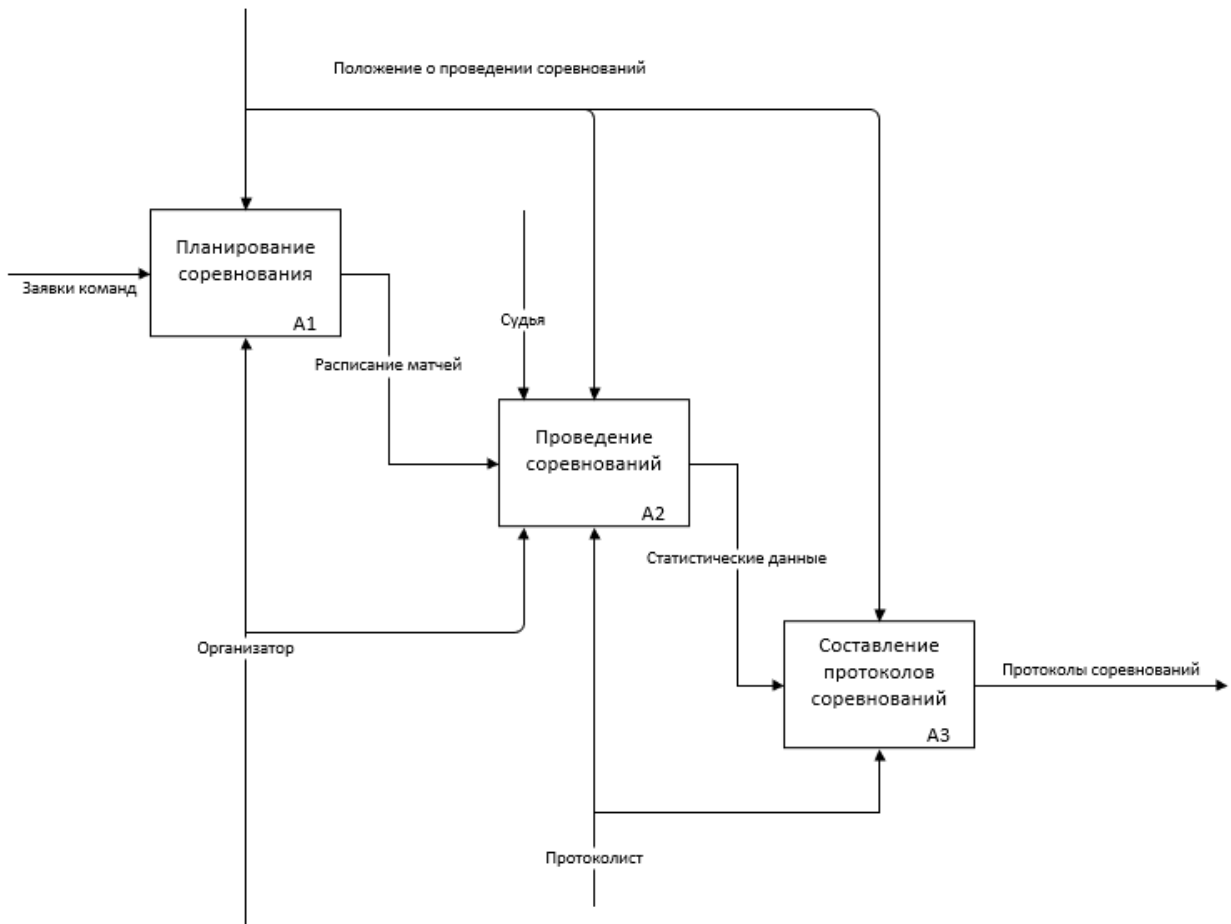


Рисунок 3.6 – Декомпозиция контекстной диаграммы.

Моделирование логики работы системы требует дальнейшей декомпозиции модулей «Планирование соревнований» и «Проведение соревнований». Модуль «Планирование соревнований» отвечает за обработку информации поступающей от команд, желающих принять участие в соревновании. Его основной функцией является составление расписания матчей, на основе полученных заявок от участников. Диаграмма данного модуля представлена на рисунке 3.4.

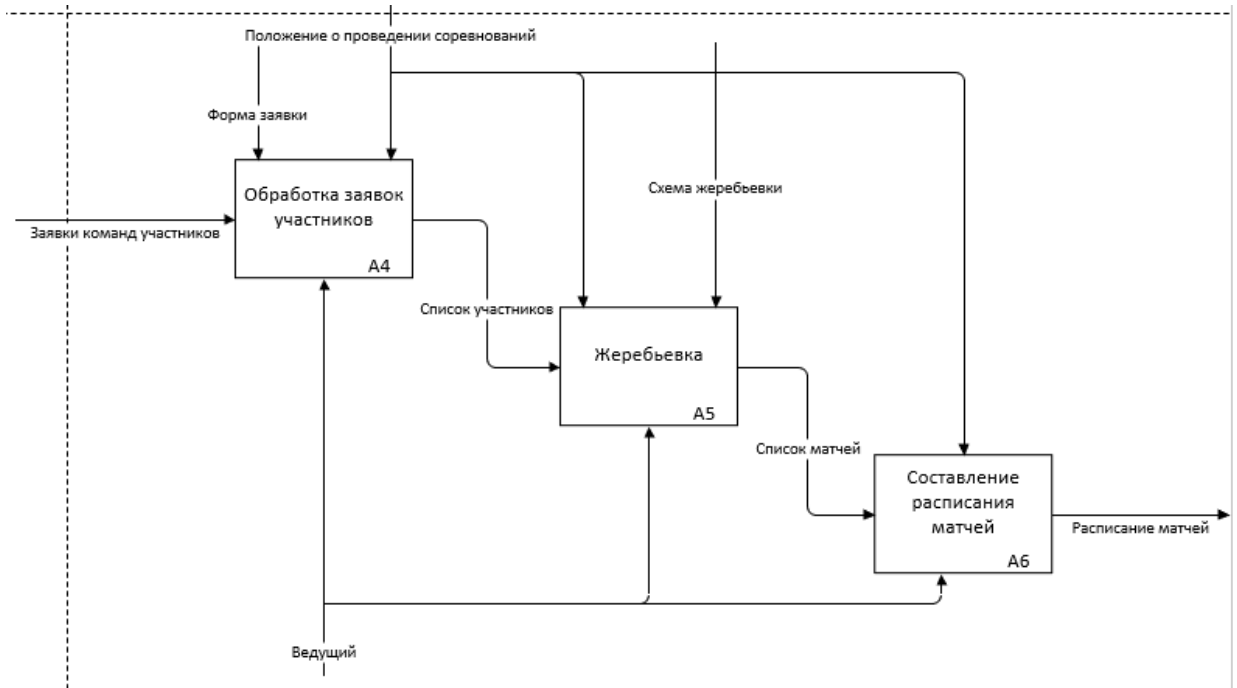


Рисунок 3.7 – Диаграмма модуля «Планирование соревнований».

Модуль «Проведение соревнований» отвечает за сбор статистических данных по ходу проведения отдельных матчей. Его основной функцией является формирование протоколов основываясь на которых система сможет выполнять аналитические функции. Диаграмма данного модуля представлена на рисунке 3.5.

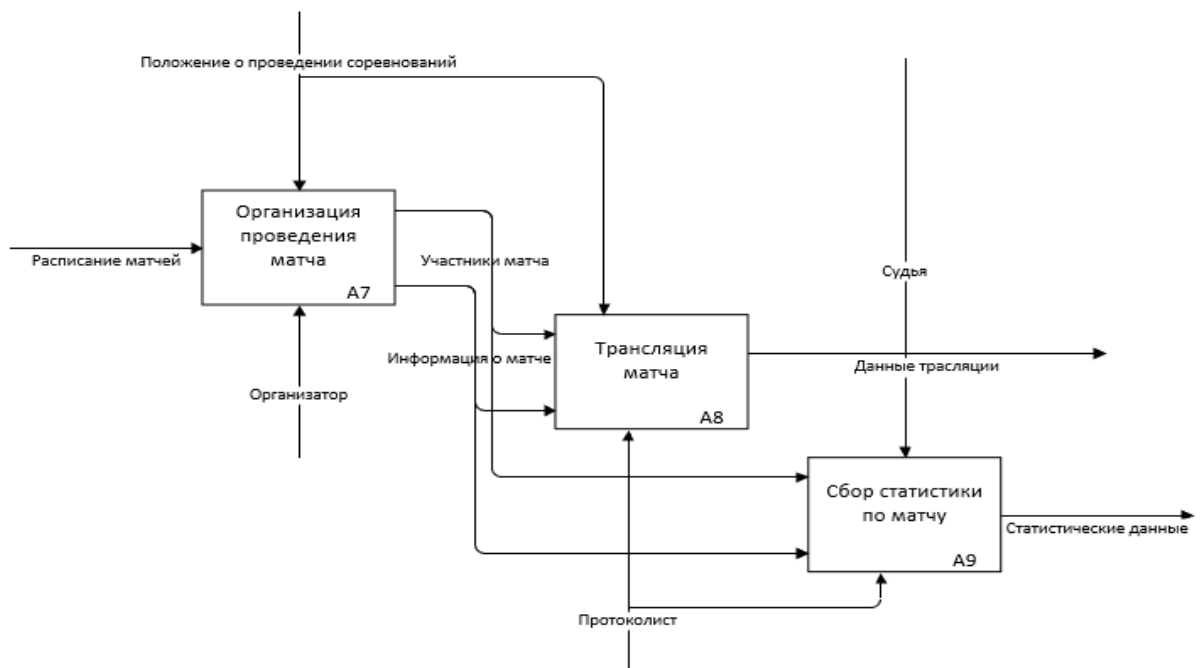


Рисунок 3.8 – Диаграмма модуля «Проведение соревнований».

Модуль «Составление протоколов соревнований» представляет собой не декомпозируемый процесс. В результате его работы статистические данные о выступлениях команд участников сохраняются в базу данных, а также формируются протоколы соревнований.

### **3.4 Макеты интерфейса**

#### **3.4.1 Пользовательский интерфейс мобильного приложения**

Пользовательский интерфейс представляет собой графическую структуру приложения. Он состоит из кнопок, нажимаемых пользователями, текстов, которые они читают, изображений, полей ввода текста и всех остальных элементов, с которыми взаимодействует пользователь. Кроме того, он включает в себя макет экрана, переходы, анимацию интерфейса и каждое микровзаимодействие. Любой визуальный элемент, взаимодействие или анимация должны быть разработаны в процессе дизайна пользовательского интерфейса.

Приложение для мобильного устройства включает в себя несколько активных экранов. Эти экраны представляют собой следующие функциональные единицы:

- экран регистрации данных;
- экран редактирования данных;
- экран сбора статистических данных матча.

Макеты интерфейсов экранов регистрации и редактирования данных представлены на рисунках 3.9 и 3.10 соответственно.

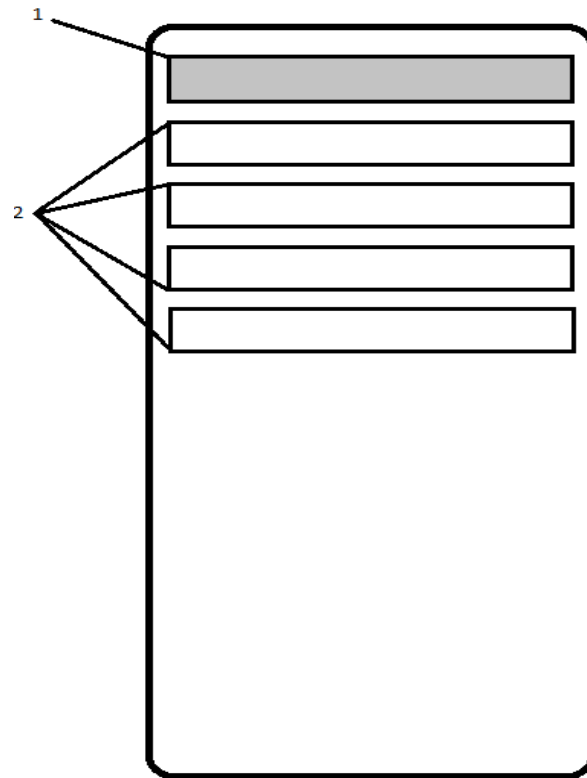


Рисунок 3.9 – Макет экрана регистрации данных; 1 – кнопка добавления данных, 2 – поля отображения введенных данных.

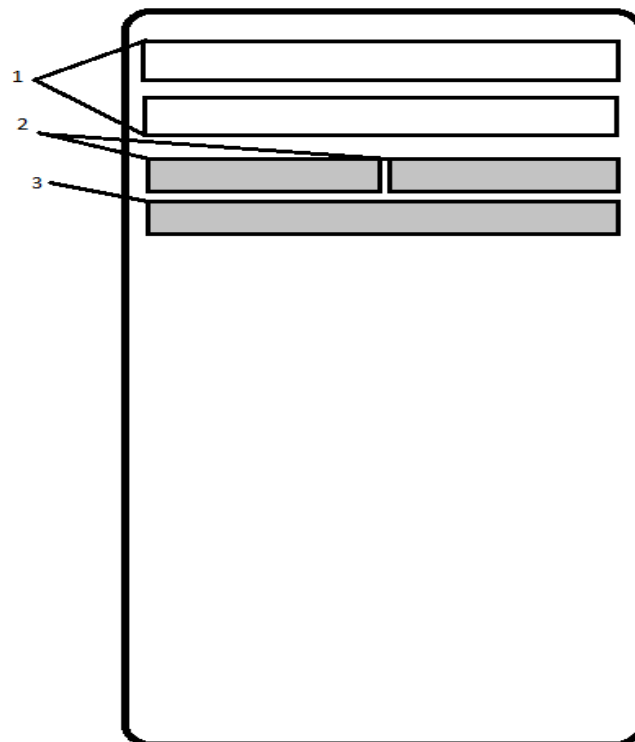


Рисунок 3.10 – Макет экрана редактирования данных; 1 – поля ввода и редактирования данных, 2 – кнопки сохранения и удаления записей, 3 – кнопка перехода к следующему экрану.

Третий экран включает в себя функционал по ведению счета, как текущего, так и общего, а также функцию отслеживания подающей стороны. Макет интерфейса, соответствующий данному функционалу представлен на рисунке 3.11.

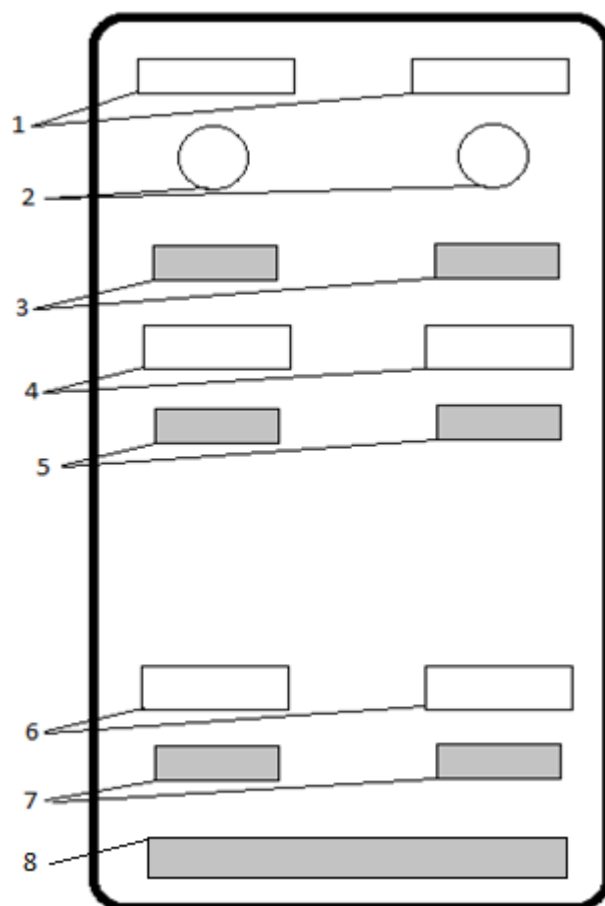


Рисунок 3.11 – Макет экрана сбора статистических данных матча; 1 – поля отображающие названия команд участников, 2 – индикатор подающей стороны, 3 – кнопки увеличения текущего счета, 4 – поля отображения текущего счета сторон, 5 – кнопки уменьшения текущего счета, 6 – поля отображения общего счета, 7 – кнопки завершения раундов сторон, 8 – кнопка завершения матча.

Представленные макеты являются заготовками для оформления функциональных экранов. Так как экраны могут быть предназначены для сбора и регистрации разных типов данных конечный их вид может незначительно отличаться от созданных макетов.



### 3.4.2 Макет веб-табло

Веб-табло предназначено для отображения статистической информации о ходе матчей проводимого соревнования. Целью его создания является трансляция актуальной информации о ходе соревнования на удаленные устройства.

Такое табло должно отображать:

- Номер текущей партии;
- Номер текущего раунда;
- Названия команд соперников;
- Счёт по раундам;
- Счёт по партиям.

Макет веб-табло, предназначенного для отображения описанной выше информации представлен на рисунке 3.12.

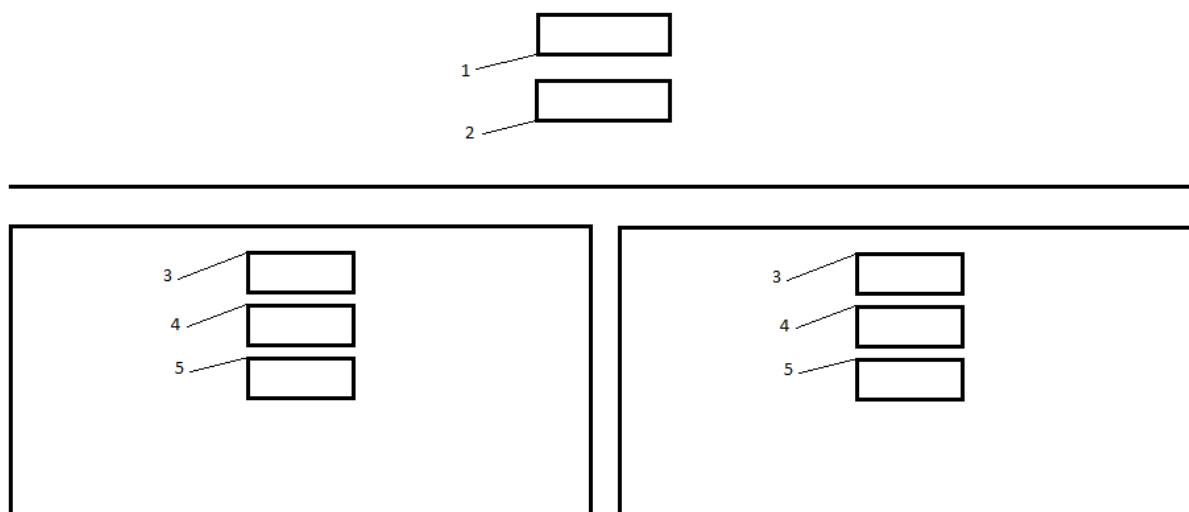


Рисунок 3.12 – Макет веб-табло; 1- Поле отображения номера партии, 2 – поле отображения номера раунда, 3 – поле отображения названия команды, 4 – поле отображения счета по раундам, 5 – поле отображения счета по партиям.

### 3.5 Создание логической модели БД

Построение логической модели БД предполагает определенную последовательность действий: определение сущностей; определение зависимостей между сущностями; определение атрибутов сущностей; задание первичных и альтернативных ключей; приведение модели к требуемому уровню нормаль-

ной формы; переход к физическому описанию модели: назначение соответствий имя сущности – имя таблицы, атрибут сущности – атрибут таблицы; задание процедур и ограничений.

Для разработки логической модели мы будем использоваться ER-модель, как самая оптимальная. Подробное описание данной модели представлено далее.

**ER- модель сущность-связь.** Модель сущность-связь (Entity Relationship Diagram (ERD)) является самым высоким уровнем в модели данных. Она определяет набор сущностей, атрибутов и взаимосвязей проектируемой системы. Модель отражает основные бизнес-правила проектируемой системы. ER-модель для проектируемой базы данных мобильного приложения приведена на рисунке 3.13.

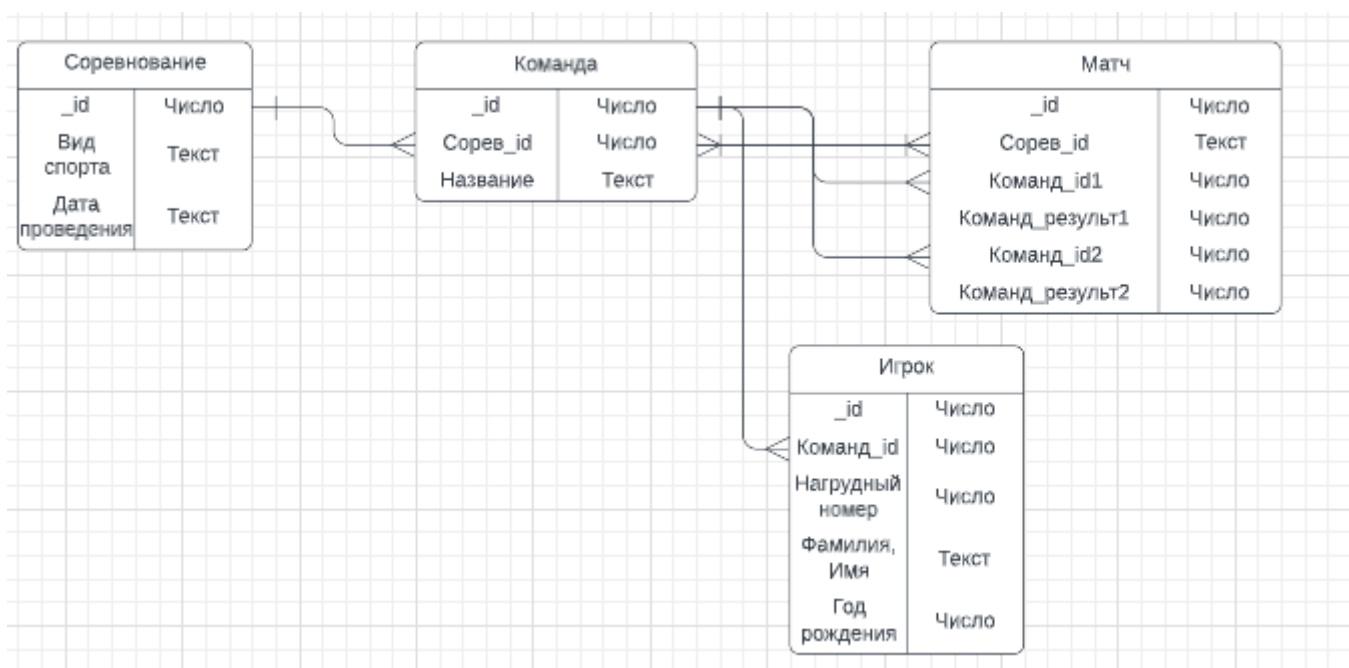


Рисунок 3.13 – Логическая модель БД (ER - модель).

Проектируемая система так же включает в себя веб-табло, для функционирования которого то же необходима база данных. Так как функционал табло предполагает только вывод данных на веб страницу, без каких-либо дополнительных операций, база данных веб-табло может иметь вид, представленный на рисунке 3.14.

Данные табло	
_id	Число
Название команды1	Текст
Название команды2	Текст
Партия	Число
Раунд	Число
Счет по раундам1	Число
Счет по партиям1	Число
Счет по раундам2	Число
Счет по партиям2	Число

Рисунок 3.14 – База данных веб-табло.

Модель этого уровня может включать связи многие-ко-многим и не включать описание ключей. Целью этой модели является формирование общего взгляда на систему для ее дальнейшей детализации. Обычно, ER-модель используется для презентаций и обсуждения структуры данных с экспертами предметной области.

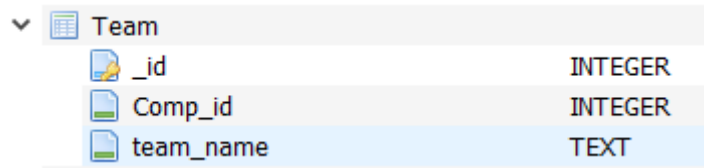
### 3.6 Создание физической структуры БД

Ранее в разделе 2.2 было принято решение использовать для разработки БД СУБД SQLite.

Построенная ранее логическая модель БД содержит всю необходимую информацию о сущностях их атрибутах и связях между ними. В среде DB Browser for SQLite сущности представляют собой таблицы, а все атрибуты сущностей являются столбцами в соответствующих таблицах. После создания всех указанных таблиц, структура базы данных имеет следующий вид (рисунки 3.14– 3.17).

▼	Competition	
	_id	INTEGER
	name	TEXT
	year	TEXT

Рисунок 3.15 – Таблица «Competition».



The screenshot shows a database table named 'Team'. It has three columns: '\_id' (INTEGER), 'Comp\_id' (INTEGER), and 'team\_name' (TEXT). The 'team\_name' column is highlighted in blue.

Column	DataType
_id	INTEGER
Comp_id	INTEGER
team_name	TEXT

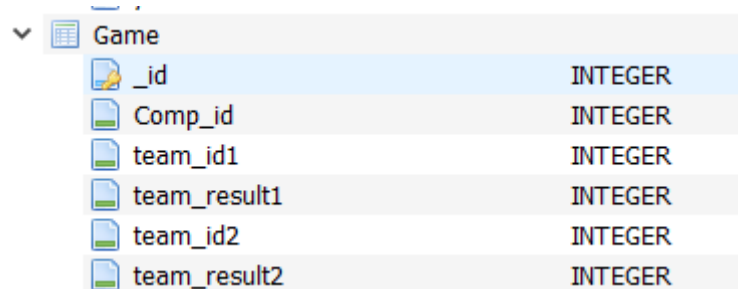
Рисунок 3.16 – Таблица «Team».



The screenshot shows a database table named 'Player'. It has five columns: '\_id' (INTEGER), 'team\_id' (INTEGER), 'number' (INTEGER), 'init' (TEXT), and 'year' (INTEGER). The '\_id' column is highlighted in blue.

Column	DataType
_id	INTEGER
team_id	INTEGER
number	INTEGER
init	TEXT
year	INTEGER

Рисунок 3.17 – Таблица «Player».



The screenshot shows a database table named 'Game'. It has seven columns: '\_id' (INTEGER), 'Comp\_id' (INTEGER), 'team\_id1' (INTEGER), 'team\_result1' (INTEGER), 'team\_id2' (INTEGER), and 'team\_result2' (INTEGER). The '\_id' column is highlighted in blue.

Column	DataType
_id	INTEGER
Comp_id	INTEGER
team_id1	INTEGER
team_result1	INTEGER
team_id2	INTEGER
team_result2	INTEGER

Рисунок 3.18 – Таблица «Game».

Связи между созданными таблицами совпадают с таковыми в разработанной в разделе 3.5 ER модели базы данных.

В разделе 2.3 было принято решение о применении СУБД MySQL для разработки серверной БД. Инструменты для взаимодействия с данной СУБД, предоставленные Open Server Panel, позволяют создать необходимые таблицы, представленные в разделе 3.5. Физическая модель базы данных веб-табло представлена на рисунке 3.19.

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно	Действие
<input type="checkbox"/>	1 id	int(11)			Нет	Нен		AUTO_INCREMENT	
<input type="checkbox"/>	2 teamName1	text	utf8mb4_unicode_ci		Нет	Нен			
<input type="checkbox"/>	3 teamName2	text	utf8mb4_unicode_ci		Нет	Нен			
<input type="checkbox"/>	4 Part	int(11)			Нет	Нен			
<input type="checkbox"/>	5 Round	int(11)			Нет	Нен			
<input type="checkbox"/>	6 scoreRound1	int(11)			Нет	Нен			
<input type="checkbox"/>	7 ScoreRound2	int(11)			Нет	Нен			
<input type="checkbox"/>	8 ScorePart1	int(11)			Нет	Нен			
<input type="checkbox"/>	9 ScorePart2	int(11)			Нет	Нен			

Рисунок 3.19 – Представление таблицы данных веб-табло.

### 3.7 Создание формы заявок и протоколов

На основе данных фигурирующих в логической и физической моделях баз данных необходимо создать форму документа, который должен содержать информацию о потенциальных участниках планируемого соревнования, а также форму документа, содержащего статистические данные прошедших в рамках соревнования матчей. Для обоих документов предполагается создание табличной структуры, так как такая структура реализует более компактную и удобную для человеческого восприятия вариацию представления данных.

Используя инструменты редактирования табличных листов MS Excel, балы создана представленная на рисунке 3.20 форма заявки на участие.

Рисунок 3.20 – Форма заявки на участие.

Форма протокола соревнований также предполагает наличие табличной структуры, содержащей следующую информацию о прошедших матчах:

- Названия противоборствующих команд;
- Состав противоборствующих команд;
- Поля ведения счета по раундам в течении партий;
- Поля результирующие командный счет по партиям;
- Поле результирующее финальный счет матча;
- Название команды победителя;
- Поля для подписей представителей команд.

Опираясь на представленную выше информацию средствами табличного редактора MS Excel была создана двухстраничная форма протокола соревнований. Так как форма протокола превышает объемы страницы формата А4, в рамках отчета она представлена частями на рисунках 3.21 – 3.23.

Протокол соревнований по волейболу					
Команда А:					
Команда А					
№	Фамилия, Имя	Нагрудный №	Год рождения		
1					
2					
3					
4					
5					
6					
7					
8					
9					
10					
11					
12					
Команда Б:					
Команда Б					
№	Фамилия, Имя	Нагрудный №	Год рождения		
1					
2					
3					
4					
5					
6					
7					
8					
9					
10					
11					
12					

Рисунок 3.21 – Часть формы протокола: «Данные о противоборствующих командах».

Партия: 1 - А		Б	
Партия: 2 - А		Б	
Партия: 3 - А		Б	
Партия: 4 - А		Б	
Партия: 5 - А		Б	
Финальный счет: А		Б	
Команда победительница			
Представитель А:			
	(подпись)		(расшифровка)
Представитель Б:			
	(подпись)		(расшифровка)

Рисунок 3.22 – Часть формы протокола «Результирующие данные матча».

Счет игры по партиям														
1			2			3			4			5		
А	№	Б	А	№	Б	А	№	Б	А	№	Б	А	№	Б
	1			1			1			1			1	
	2			2			2			2			2	
	3			3			3			3			3	
	4			4			4			4			4	
	5			5			5			5			5	
	6			6			6			6			6	
	7			7			7			7			7	
	8			8			8			8			8	
	9			9			9			9			9	
	10			10			10			10			10	
	11			11			11			11			11	
	12			12			12			12			12	
	13			13			13			13			13	
	14			14			14			14			14	
	15			15			15			15			15	
	16			16			16			16			16	
	17			17			17			17			17	
	18			18			18			18			18	
	19			19			19			19			19	
	20			20			20			20			20	
	21			21			21			21			21	
	22			22			22			22			22	
	23			23			23			23			23	
	24			24			24			24			24	
	25			25			25			25			25	
	26			26			26			26			26	
	27			27			27			27			27	
	28			28			28			28			28	
	29			29			29			29			29	
	30			30			30			30			30	

Рисунок 3.22 – Часть формы протокола: «Счет по раундам в течении партий».

Будучи заполненной, созданная форма протокола является выходным потоком данных согласно представленной в разделе 3.3 схеме системы.



## 4 Разработка программной системы

### 4.1 Создание интерфейса мобильного приложения

Интерфейс Android приложения при разработке в Android studio формируется с помощью языка разметки XML. XML – расширяемый язык разметки, предназначенный для хранения и передачи данных. Документ XML состоит из элементов, начинающихся открывающим тегом в угловых скобках, затем идет содержимое элемента, после него записывается закрывающий тег в угловых скобках. Пример кода, отвечающего за отображение списка элементов и пример кода отвечающего за отображение кнопки представлены на рисунках 4.1 и 4.2 соответственно.

```
<ListView
    android:id="@+id/list"
    android:layout_width="0dp"
    android:layout_height="0dp"
    app:layout_constraintTop_toBottomOf="@+id/addButton"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintRight_toRightOf="parent"/>
```

Рисунок 4.1 – Код отображения списка элементов.

```
<Button
    android:id="@+id/addButton"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:textSize="18sp"
    android:text="Добавить соревнование"
    android:onClick="add"
    app:layout_constraintBottom_toTopOf="@+id/list"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintLeft_toLeftOf="parent"
    app:layout_constraintRight_toRightOf="parent"
/>
```

Рисунок 4.2 – Код отображения кнопки.

Все элементы интерфейса, необходимые для реализации функционала, размещаются на экране в соответствии с макетом, представленным в разделе

3.4. Программная модель интерфейса, разработанная в Android studio, для окна сбора статистических данных матчей, представлена на рисунке 4.3.

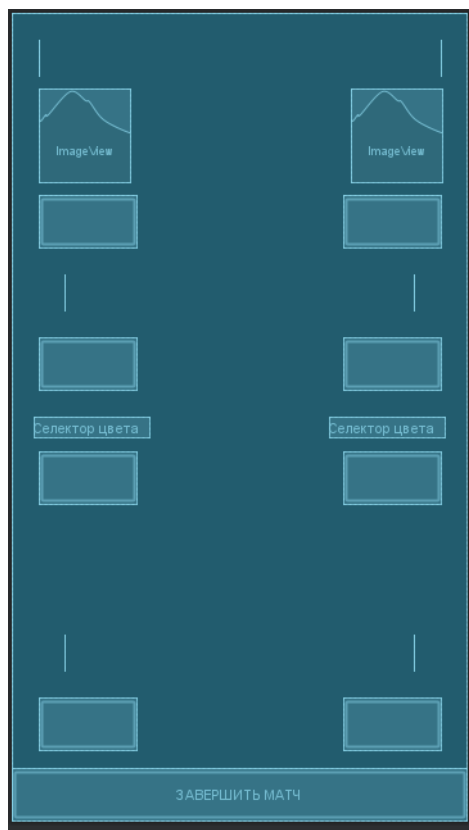


Рисунок 4.3 – Представление интерфейса экрана сбора статистических данных матча.

В ходе разработки были созданы несколько экранов регистрации и редактирования данных. Они отличаются друг от друга количеством и названиями полей ввода и вывода данных, однако каждый разработанный экран придерживается шаблона, представленного в разделе 3.4.

## 4.2 Взаимодействие с базой данных

В разрабатываемом android приложении применяется СУБД SQLite, описанная ранее в разделе 2.2. Взаимодействие приложения с базой данных осуществляется через встроенную библиотеку SQLite. Методы SQLite предполагают прямое обращение к файлам базы данных и коммуникацию с ними через язык структурированных запросов SQL. При прямом вшивании SQL запросов в код разрабатываемого приложения возникает проблема, требующая

множественного редактирования кода в случае внесения изменений в структуру базы данных. Для решения данной проблемы в ходе разработки необходимо создавать вспомогательные классы, которые будут хранить информацию о структуре базы данных и помогать с формированием SQL запросов. Пример кода, хранящего информацию о базе данных и осуществляющего доступ к ней представлены на рисунках 4.4 и 4.5.

```
private static String DB_PATH; // полный путь к базе данных
private static String DB_NAME = "VLDB4.db";
private static final int SCHEMA = 1; // версия базы данных
static final String TABLE = "Competition"; // название таблицы в бд
// названия столбцов
static final String COLUMN_ID = "_id";
static final String COLUMN_NAME = "name";
static final String COLUMN_YEAR = "year";
private Context myContext;
```

Рисунок 4.4 – Основные данные связанной с вспомогательным классом таблицы.

```
void create_db(){
    File file = new File(DB_PATH);
    if (!file.exists()) {
        //получаем локальную бд как поток
        try(InputStream myInput = myContext.getAssets().open(DB_NAME);
            // Открываем пустую бд
            OutputStream myOutput = new FileOutputStream(DB_PATH)) {

            // побайтово копируем данные
            byte[] buffer = new byte[1024];
            int length;
            while ((length = myInput.read(buffer)) > 0) {
                myOutput.write(buffer, off: 0, length);
            }
            myOutput.flush();
        }
        catch(IOException ex){
            Log.d( tag: "DatabaseHelper", ex.getMessage());
        }
    }
}

public SQLiteDatabase open()throws SQLException {

    return SQLiteDatabase.openDatabase(DB_PATH, factory: null, SQLiteDatabase.OPEN_READWRITE);
}
```

Рисунок 4.5 – Методы доступа к базам данных.

При помощи вспомогательных классов SQL запросы формируются с использованием ссылок на данные, сохраненные в этих классах. Пример такого SQL запроса представлен на рисунке 4.6.

```
userCursor = db.rawQuery( sql: "select " + DatabaseHelperG.TABLE + "." + DatabaseHelperG.COLUMN_ID + ","
+ DatabaseHelperG.TABLE + "." + DatabaseHelperG.COLUMN_TID1 + ","
+ DatabaseHelperG.TABLE + "." + DatabaseHelperG.COLUMN_RESULT1 + "," + DatabaseHelperT.TABLE + "." + DatabaseHelperT.COLUMN_NAME
+ " from " + DatabaseHelperG.TABLE
+ " Join " + DatabaseHelperT.TABLE + " on " + DatabaseHelperT.TABLE + "." + DatabaseHelperT.COLUMN_ID + " = "
+ DatabaseHelperG.TABLE + "." + DatabaseHelperG.COLUMN_TID1, selectionArgs: null);
```

Рисунок 4.6 – Пример SQL запроса.

Для внесения изменений в запросы, формируемые подобным образом, редактирование кода необходимо осуществлять только в области представленной на рисунке 4.6. Результаты запросов записываются в курсоры. Вместо того чтобы извлекать данные и возвращать копию значений, курсоры ссылаются на результирующий набор исходных данных. Курсоры позволяют управлять текущей позицией (строкой) в результирующем наборе данных, возвращаемом при запросе. Заключительным взаимодействием данной цепочки является вывод результирующего набора данных в список элементов.

Аналогичным образом осуществляется взаимодействие с каждой отдельно взятой таблицей базы данных.

### 4.3 Реализация функционала мобильного приложения

В процессе разработки мобильного приложения требуется реализовать следующий функционал:

- ввод данных протоколиста;
- добавление планируемого соревнования;
- добавление команд участников;
- добавление игроков команд;
- получение списка матчей;
- сбор статистических данных матча;
- формирование протоколов матчей;
- обзор результатов проведенных матчей.

Экран ввода данных протоколиста является стартовым экраном приложения. Он представляет собой простую форму, содержащую поля ввода фамилии, имени, отчества и кнопку подтверждающую ввод информации. Интерфейс данного экрана представлен на рисунке 4.7.

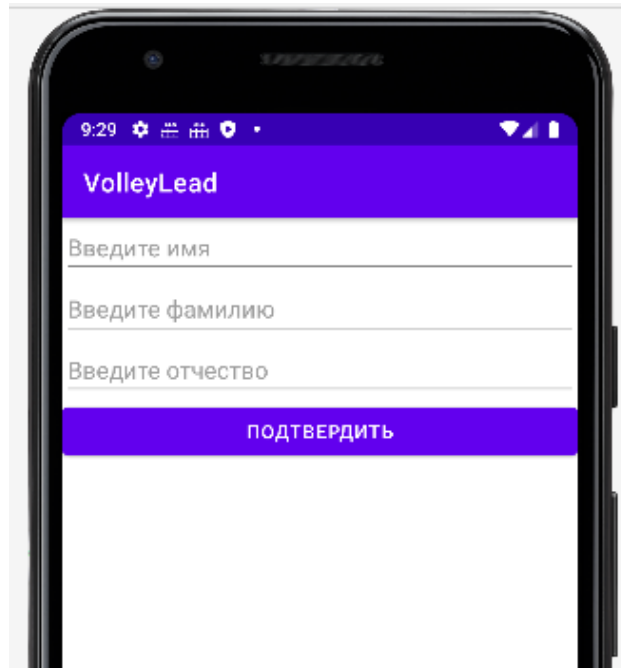


Рисунок 4.7 – Интерфейс стартового экрана приложения.

После подтверждения действия на стартовом экране приложение предоставляет пользователю доступ к следующей форме, отвечающей за добавление в список планируемого соревнования. Интерфейс экрана добавления соревнования представлен на рисунке 4.8.

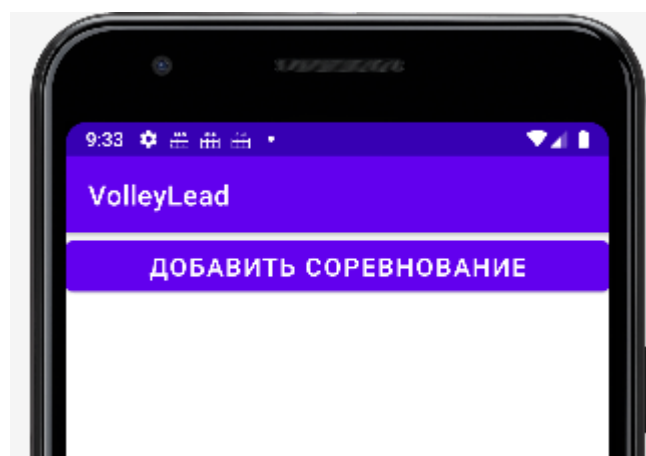


Рисунок 4.8 – Интерфейс экрана добавления соревнования.

Взаимодействие с кнопкой «Добавить соревнование» открывает доступ к промежуточному экрану, отвечающему за занесение записей о планируемом соревновании в базу данных. Интерфейс данного промежуточного экрана представлен на рисунке 4.9.

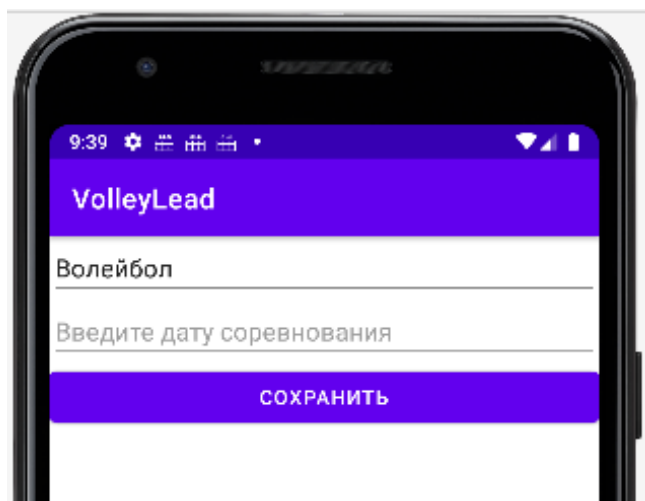


Рисунок 4.9 – Интерфейс экрана редактирования записей о соревнованиях.

По мере добавления в базу данных записей о планируемых соревнованиях на экране, представленном на рисунке 4.8 будут отображаться элементы, взаимодействие с которыми откроет пользователю доступ к следующему экрану, отвечающему за добавление и отображение команд участников соревнований, а также за переход к экрану создания расписания матчей. Интерфейс данного экрана представлен на рисунке 4.10.

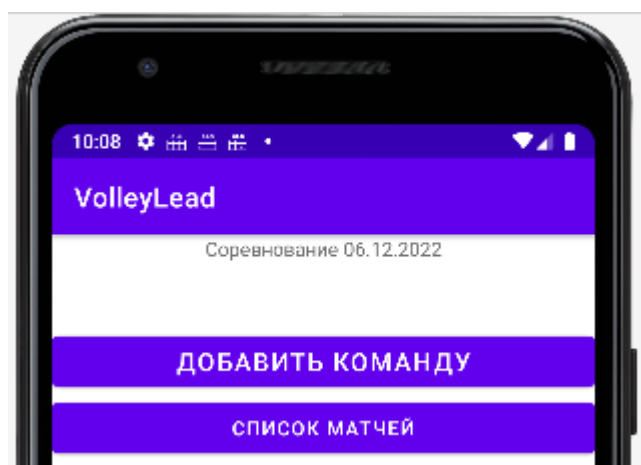


Рисунок 4.10 – Интерфейс экрана добавления команд.

Взаимодействие с кнопкой «Добавить команду» открывает пользователю доступ к промежуточному экрану, отвечающему за внесение информации о командах участников в базу данных. Интерфейс данного промежуточного экрана представлен на рисунке 4.11.

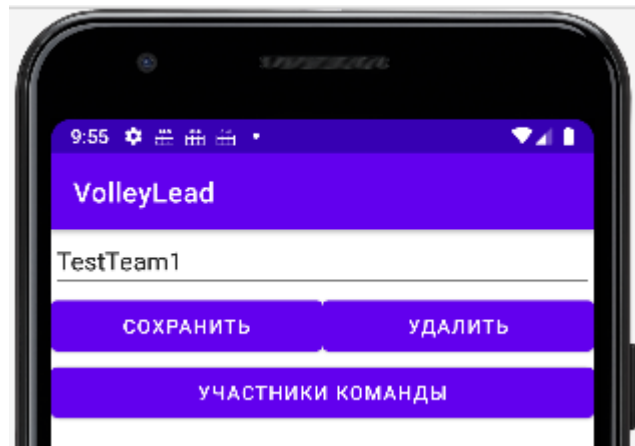


Рисунок 4.11 – Интерфейс экрана редактирования записей о командах.

Из экрана, предназначенного для редактирования записей о командах участников, пользователь может перейти к экрану отображения и добавления записей о игроках команд, путем взаимодействия с кнопкой «Участники команды». Интерфейс данного окна представлен на рисунке 4.12.

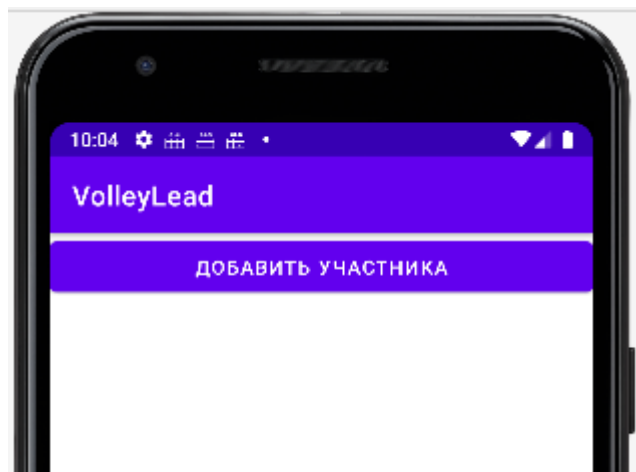


Рисунок 4.12 – Интерфейс окна добавления записей о игроках.

Взаимодействуя с кнопкой «Добавить участника», пользователь получает доступ к промежуточному экрану, отвечающему за внесение информации о игроках целевой команды в базу данных. Интерфейс экрана редактирования записей о игроках представлен на рисунке 4.13

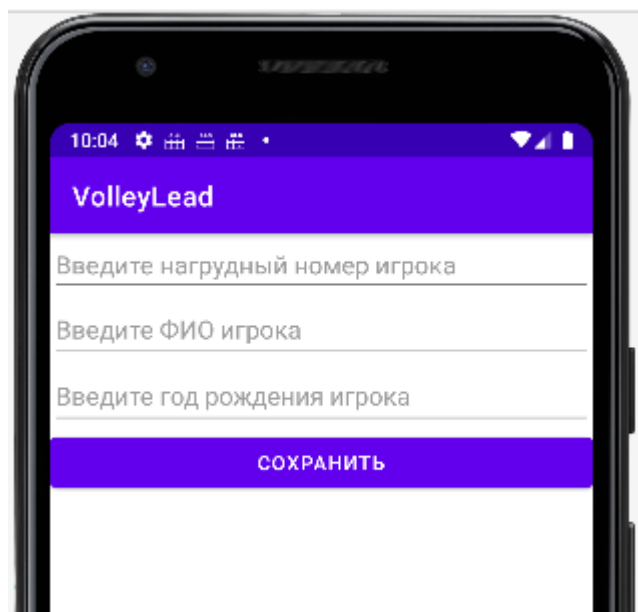


Рисунок 4.13 – Интерфейс экрана редактирования записей о игроках.

Таким образом завершается функциональная ветка внесения всей необходимой информации о командах участниках соревнования в базу данных. После этого пользователь, вернувшись к экрану, представленному на рисунке 4.10 получает возможность сформировать список матчей соревнования. Для этого пользователю необходимо взаимодействовать с кнопкой «Список матчей». Составление такого списка осуществляется на основе информации о командах участниках, введенной ранее, путем обработки строковых списков, методами ArrayList. ArrayList предоставляет расширенный функционал для работы с массивами. Он реализует такие функции как автоматическое переопределение объема необходимой памяти, добавление и удаление элементов из массива, вставка элементов на требуемые позиции. Пример кода, отвечающего за формирование списка матчей представлен на рисунке 4.14.



```

tv = findViewById(R.id.textView2);
tv.setText("Матчи " + date);
tempt = new ArrayList<>();
tempt2 = new ArrayList<>();
schedule = new ArrayList<>();
Collections.addAll(tempt, teams);
Collections.addAll(tempt2, teams);
while (tempt.size()>0){
    for(int i = 0; i < tempt2.size(); i++){
        String temp = tempt.get(0);
        String temp2 = tempt2.get(i);
        if (temp != temp2){
            String temp3 = temp + " против " +temp2;
            schedule.add(temp3);
        }
    }
    tempt.remove(index: 0);
    tempt2.remove(index: 0);
}
end = new String[schedule.size()];
schedule.toArray(end);
ArrayAdapter<String> adapter = new ArrayAdapter(context: this,
        android.R.layout.simple_list_item_1, end);
userList.setAdapter(adapter);

```

Рисунок 4.14 – функция формирования списка матчей.

Полученный в результате массив строк обрабатывается и выводится на экран. Пример интерфейса экрана формирования списка матчей соревнования представлен на рисунке 4.15.

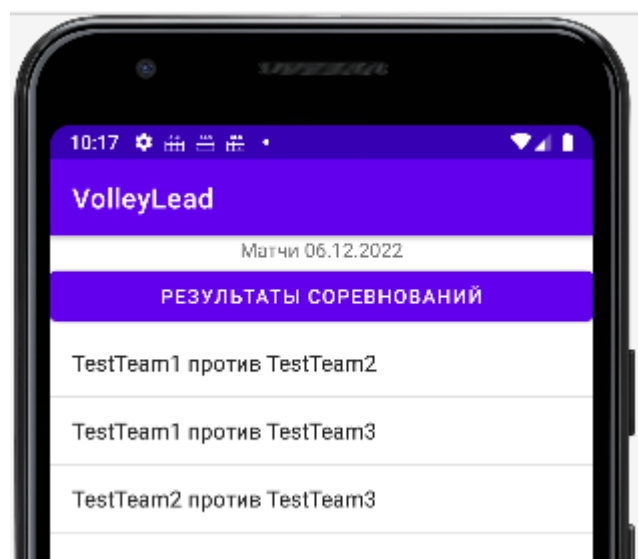


Рисунок 4.15 – Интерфейс экрана формирования списка матчей.

Взаимодействие с элементами списка матчей позволяют перейти экрану разрабатываемого приложения, экрану, отвечающему за сбор статистических данных проводимого матча. Данный экран позволяет фиксировать текущий счет раунда, отслеживать подающую команду, а также, за счет функции установки цвета формы, облегчает идентификацию противоборствующих команд. Изменение счетчика очков в текстовых полях происходит за счет применения к числовой переменной унарных операторов. Увеличивающих или уменьшающих значения переменных на единицу по нажатию соответствующей кнопки. Пример кода отвечающего за обработку нажатия кнопки увеличения количества очков команды представлен на рисунке 4.16.

```
public void scorePlus1(View view) {
    scoreT1 ++;
    String scoreT1S = Integer.toString(scoreT1);
    TextView textView1 = findViewById(R.id.textView);
    textView1.setText(scoreT1S);
    imageView.setVisibility(View.VISIBLE);
    imageView2.setVisibility(View.INVISIBLE);
}
```

Рисунок 4.16 – Обработка нажатия кнопки «+».

Те же функции так же отвечают за отслеживание подающей стороны путём переключения режимов видимости соответствующих элементов.

Помимо описанного выше в рамках данного экрана также реализуется функционал формирования протоколов матчей на основе созданной в разделе 3.7 формы протокола. Так как положение полей данных в форме протокола известно заранее, появляется возможность точечного заполнения документа «основы» новыми данными, собираемыми в ходе матчей. Для реализации данной возможности в рамках языка программирования Java необходимо использовать стороннюю библиотеку, позволяющую осуществлять работу с файлами типов .xls и .xlsx.

Такой библиотекой является Apache POI. Apache POI представляет собой API, который позволяет использовать файлы MS Excel в приложениях со-

зданных с помощью Java. Данная библиотека является свободно распространяемой. Она включает в себя классы и методы для чтения и записи информации в документы MS Office, в том числе и в документы MS Excel [11].

Пример программного кода, демонстрирующий работу с методами и классами библиотеки Apache POI представлен на рисунке 4.17.

```
path = getAssets().open( fileName: "Форма протокола.xls");
myProtocol = new HSSFWorkbook(path);
myPSheet = myProtocol.getSheetAt( index: 0);
row = myPSheet.getRow( rowIndex: 2);
tname = row.getCell( cellnum: 1);
tname.setCellValue(teams[0]);
row = myPSheet.getRow( rowIndex: 20);
tname = row.getCell( cellnum: 1);
tname.setCellValue(teams[2]);
row = myPSheet.getRow( rowIndex: 51);
tname = row.getCell( cellnum: 2);
tname.setCellValue(date);
row = myPSheet.getRow( rowIndex: 50);
tname = row.getCell( cellnum: 2);
tname.setCellValue(prName);
path.close();
out = new FileOutputStream( name: way + "P_" + date + "_" + teams[0] + "_" + teams[2] + ".xls");
myProtocol.write(out);
out.close();
```

Рисунок 4.17 – Добавление данных в форму протокола.

Как видно по рисунку 4.17 использование библиотеки Apache POI позволяет добавлять различные данные в любую часть табличной сетки, ссылаясь на номера требуемых строки и столбца.

Третьим крупным функциональным блоком, реализованным в рамках экрана сбора и сохранения статистических данных, является функционал передачи данных на веб-табло. Программный код реализующий описанный ранее функционал будет представлен в следующих разделах.

Для наглядности примером интерфейса экрана сбора и сохранения статистических данных матча послужит экран уже наполненный тестовыми данными. Данный пример представлен на рисунке 4.18.

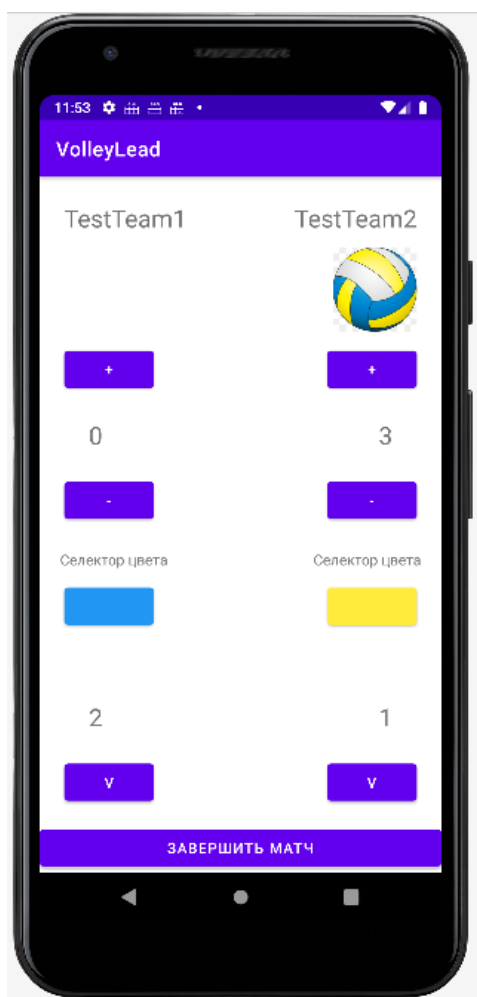


Рисунок 4.18 – Интерфейс экрана сбора и сохранения данных матча.

Взаимодействие пользователя с кнопкой «Завершить матч» инициирует два завершающих действия. Первым действием является завершение формирования протокола матча, при котором в протокол заносятся финальный счет и название команды победителя. Вторым действием является занесение в базу данных записей финальных счетов команд участников завершеного матча.

Далее пользователь имеет возможность взаимодействовать с кнопкой «Результаты соревнований», представленной на рисунке 4.15. Данное взаимодействие предоставит пользователю доступ к экрану обзора результатов матчей. Пример интерфейса данного окна представлен на рисунке 4.19.

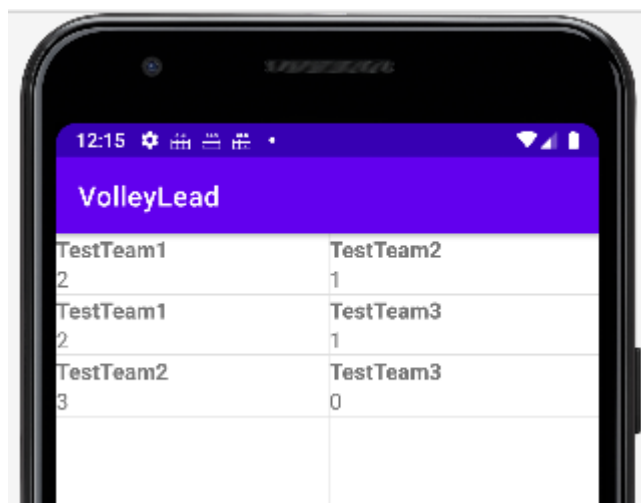


Рисунок 4.19 – Интерфейс экрана обзора результатов матчей

Если пользователь принимает решение о необходимости выполнения каких-либо действий с фалом протокола завершенного соревнования, то пользователю необходимо перейти в файловый проводник мобильного устройства. Файл протокола будет находиться в общей файловой директории внутреннего хранилища мобильного устройства. Название файла формируется в процессе занесения в него данных и принимает такой вид – «DownloadP\_Дата соревнования\_Название команды соперника1\_Название команды соперника2.xls».

#### 4.4 Разработка веб-табло

Для создания такой структуры как веб-табло используются следующие компоненты:

- html и css – создание внешнего вида веб-табло;
- php – создание выполняемых скриптов для веб-взаимодействий;
- Open Server Panel и MySQL – тестирование.

HTML – стандартизированный язык разметки документов в интернет пространстве. Большинство веб-страниц содержат описание разметки на языке HTML. Данный язык автоматически интерпретируется браузерами, а полученный в результате такой интерпретации форматированный текст отображается на экране используемого устройства [12]. Основная структурная единица языка HTML – тег. С помощью тегов браузер узнает, как правильно интерпретировать ту или иную часть текста, созданную с использованием

HTML. Однако одного HTML недостаточно для полноценного форматирования внешнего вида создаваемой веб-страницы, при необходимости добавления на веб-страницу неких визуальных решений в связке с HTML применяют также CSS.

CSS – каскадные таблицы стилей, применяемые для стилевых описаний различных HTML тегов [13]. CSS используется для определения правил оформления документов затрагивая дизайн и вёрстку. У такого способа форматирования имеется несколько достоинств[14]:

- Внешний вид всего сайта можно изменить централизованно, а вносить коррективы в форматирование каждой отдельной странички;
- Документ проще обслуживать;
- В тегах не дублируются описания параметров стилей.

Применяя описанные выше технологии была создана веб-страница, соответствующая макету, представленному в разделе 3.4.2. Соответствующий макету код на языках HTML и CSS представлены на рисунке 4.20 и 4.21 соответственно.

```

15 <body>
16 <p align="center"><big><big><big>Партия</big></big></big></p>
17 <p align="center"><big><big>
18 </big></big></p>
19 <p align="center"><big><big><big>Раунд</big></big></big></p>
20 <p align="center"><big><big>
21 </big></big></p>
22 <hr>
23 <div id="block1">
24 <p align="center"><big><big><big>Команда</big></big></big></p>
25 <p align="center"><big><big>
26 </big></big></p>
27 <p align="center"><big><big><big>Счет по раундам</big></big></big></p>
28 <p align="center"><big><big>
29 </big></big></p>
30 <p align="center"><big><big><big>Счет по партиям</big></big></big></p>
31 <p align="center"><big><big>
32 </big></big></p>
33 </div>
34 <div id="block2">
35 <p align="center"><big><big><big>Команда</big></big></big></p>
36 <p align="center"><big><big>
37 </big></big></p>
38 <p align="center"><big><big><big>Счет по раундам</big></big></big></p>
39 <p align="center"><big><big>
40 </big></big></p>
41 <p align="center"><big><big><big>Счет по партиям</big></big></big></p>
42 <p align="center"><big><big>
43 </big></big></p>
44 </div>
45 </body>

```

Рисунок 4.20 – HTML разметка веб-табло.

```

1
2  #block1 {
3    float: left;
4    display: block;
5
6    width: 49%;
7    border: 1px solid blue;
8    margin: 2px;
9  }
10
11 #block2 {
12   float: right;
13   display: block;
14   width: 49%;
15   border: 1px solid blue;
16   margin: 2px;
17 }

```

Рисунок 4.21 – Таблица стилей веб-табло.

Результат обработки представленного выше программного кода, в среде браузера Google Chrome отображен на рисунке 4.22.

Партия	
Раунд	
Команда	Команда
Счет по раундам	Счет по раундам
Счет по партиям	Счет по партиям

Рисунок 4.22 – Внешний вид веб-табло.

Для того чтобы веб-табло функционировало необходимо передавать собираемые мобильным приложением статистические данные в базу данных веб-табло, представленную в разделе 3.6. Оптимальным инструментом для реализации данного процесса является PHP.

PHP – это распространенный язык программирования общего назначения с открытым исходным кодом. PHP специально разработан для применения его в сфере веб-разработок, благодаря чему его код может быть внедрен непосредственно в HTML[12]. В рамках разрабатываемой программной системы скрипты, написанные на PHP, отвечают, как за получение данных из мобильного приложения, так и за вывод этих данных непосредственно на веб-табло. Пример кода мобильного приложения, отвечающего за отправку данных на веб-табло представлен на рисунке 4.23.

```

URL url = new URL(myURL);
URLConnection conn = (URLConnection) url.openConnection();
conn.setRequestMethod("POST");
conn.setDoOutput(true);
conn.setDoInput(true);
conn.setRequestProperty("Content-Length", "" + Integer.toString(parameters.getBytes().length));
OutputStream os = conn.getOutputStream();
data = parameters.getBytes( charsetName: "UTF-8");
os.write(data);
data = null;
conn.connect();
int responseCode= conn.getResponseCode();
ByteArrayOutputStream baos = new ByteArrayOutputStream();
if (responseCode == 200) {
    is = conn.getInputStream();
    byte[] buffer = new byte[8192];
    int bytesRead;
    while ((bytesRead = is.read(buffer)) != -1) {
        baos.write(buffer, off: 0, bytesRead);
    }
    data = baos.toByteArray();
    resultString = new String(data, charsetName: "UTF-8");
}

```

Рисунок 4.23 – Функция отправляющая данные на сервер.

Представленный выше код осуществляет соединение с сервером по URL адресу, после чего отправляет на сервер данные в виде POST запроса, затем читает пришедший от сервера ответ.

POST запрос – один из многих видов запроса применяемый в интернет пространстве. Предназначен для на сервер данных заключенных в тело данного запроса.

Чтобы обработать полученные таким образом данные, на сервере должен присутствовать файл с PHP скриптом, осуществляющим разбор данных из POST запроса и занесение таких данных в серверную БД. Пример такого PHP представлен на рисунке 4.24.



```

<?
$hostname = "localhost";
$dbName = "table";
$username = "Table";
$password = "!234";
$conn = mysqli_connect($hostname,$username,$password, $dbName) OR DIE("No connection");
/* выбрать базу данных с ранее заданным именем. Если произойдет ошибка - вывести ее */
if (!$conn) {
    echo "Error: Impossible to connect." . PHP_EOL;
    echo "Error code errno: " . mysqli_connect_errno() . PHP_EOL;
    echo "Error text error: " . mysqli_connect_error() . PHP_EOL;
    exit;
}
echo "Connection established";
$sql = mysqli_query($conn, "INSERT INTO tabledata (teamName1, teamName2, Part, Round,
scoreRound1, scoreRound2, scorePart1, scorePart2)
VALUES ('{$_POST['teamName1']}', '{$_POST['teamName2']}', {$_POST['Part']}, {$_POST['Round']},
{$_POST['scoreRound1']}, {$_POST['scoreRound2']},
{$_POST['scorePart1']}, {$_POST['scorePart2']})");
if ($sql) {
    echo '<p>Data successfully added.</p>';
} else {
    echo '<p>Error: ' . mysqli_error($conn) . '</p>';
}
$conn->close();
?>

```

Рисунок 4.24 – Скрипт записи данных в БД.

Представленный выше код осуществляет соединение с серверной БД под управлением MySQL, созданной с помощью инструментария Open Server Panel. В случае успешного установления соединения данный скрипт сформирует из полученных в POST запросе данных SQL запрос, который поместит эти данные в соответствующую таблицу базы данных.

Чтобы получить возможность отобразить полученные из мобильного приложения данные на веб-табло требуется модифицировать представленную на рисунке 4.20 HTML разметку. В данном случае нужно воспользоваться тем фактом, что скрипт PHP может быть встроен непосредственно в HTML код. С помощью добавления, представленного на рисунке 4.25 PHP скрипта в разметку HTML, производится получение данных из серверной БД и подготовка этих данных для отображения их непосредственно на веб-табло.

```

<?php
$hostname = "localhost";
$dbName = "table";
$username = "Table";
$password = "!234";
$conn = mysqli_connect($hostname,$username,$password, $dbName) OR DIE("No connection");
if (!$conn) {
    echo "Error: Impossible to connect." . PHP_EOL;
    echo "Error code errno: " . mysqli_connect_errno() . PHP_EOL;
    echo "Error text error: " . mysqli_connect_error() . PHP_EOL;
    exit;
}
//echo "Connection established";
$sql = "SELECT * FROM tabledata ORDER BY id DESC LIMIT 1";
if($result = $conn->query($sql)){
    foreach($result as $row){
        $teamName1 = $row["teamName1"];
        $teamName2 = $row["teamName2"];
        $Part = $row["Part"];
        $round = $row["Round"];
        $scoreRound1 = $row["scoreRound1"];
        $scoreRound2 = $row["ScoreRound2"];
        $scorePart1 = $row["ScorePart1"];
        $scorePart2 = $row["ScorePart2"];
    }
}
$conn->close();
?>

```

Рисунок 4.25 – Скрипт получения данных из БД.

Полученные таким образом данные необходимо отобразить в полях веб-табло. В этом вновь помогает функция встраивания PHP кода непосредственно в разметку HTML. Пример кода отображающего полученные из серверной БД данные представлен на рисунке 4.26.

```

<p align="center"><big><big><big>Партия</big></big></big></p>
<p align="center"><big><big>
<?php
    echo "$Part";
?>
</big></big></p>
<p align="center"><big><big><big>Раунд</big></big></big></p>
<p align="center"><big><big>
<?php
    echo "$round";
?>
</big></big></p>

```

Рисунок 4.26 – Код отображающий данные из БД.

Таким образом был реализован весь функционал необходимый для функционирования веб-табло. Реализованный функционал включает в себя –

конструктор внешнего вида веб-табло, функции занесения данных в серверную БД, функции извлечения данных из серверной БД, функции вывода данных в рабочее пространство веб-табло.

#### 4.5 Тестирование системы

Чтобы убедиться, что все функции разработанной системы соответствуют исходным функциональным требованиям, необходимо провести функциональное тестирование типа «черный ящик». Такой вид тестирования подразумевает отсутствие доступа к программному коду системы в процессе тестирования. Для тестирования разрабатываемой системы оптимальным является «РФТ» или ручное функциональное тестирование. Такой тип тестирования предполагает следующий алгоритм действий:

- описание тестовой ситуации;
- воспроизведение ситуации в тестируемой системе;
- документирование результата.

Тестовая ситуация №1 – добавление и редактирование пользователем записей в базу данных. Результат выполнения тестовой ситуации №1 представлен на рисунке 4.27.

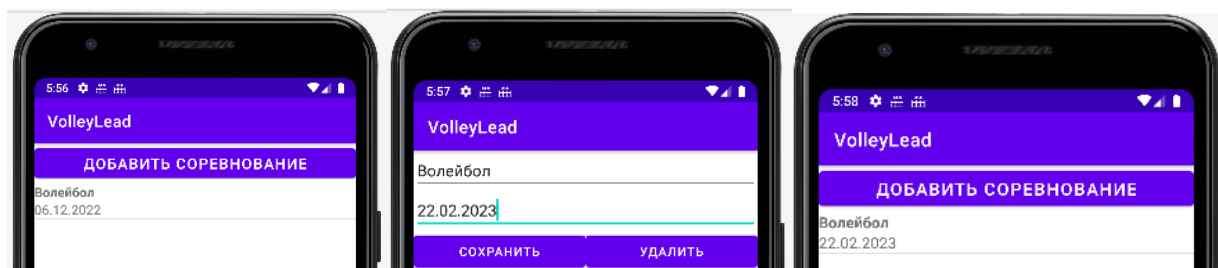


Рисунок 4.27 – Выполнение тестовой ситуации №1.

Результат – записи добавляются, сохраняются, редактируются, конфликтов в процессе выполнения не возникает.

Тестовая ситуация №2 – получение пользователем списка матчей соревнования с различным количеством участников. Результат выполнения тестовой ситуации №2 представлен на рисунке 4.28.

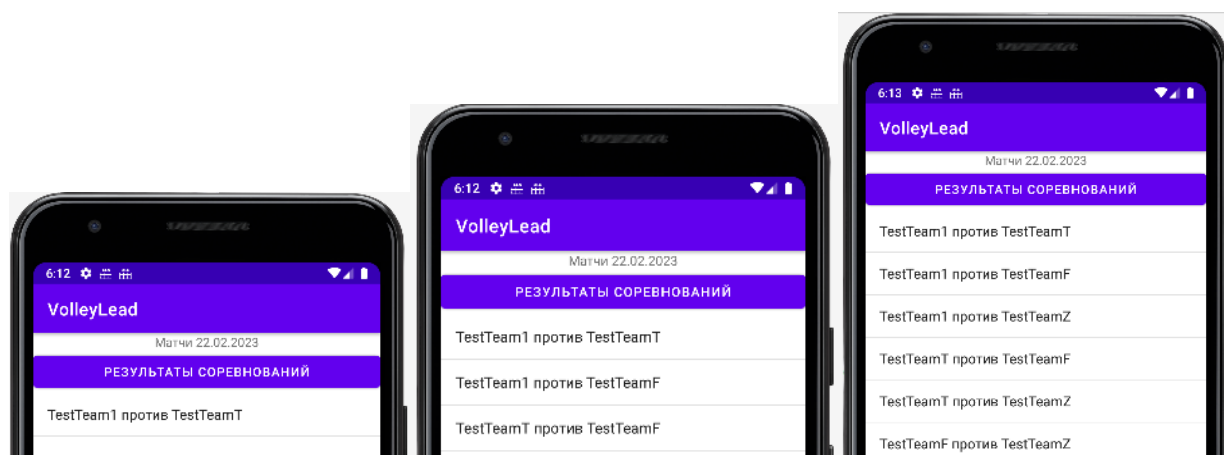


Рисунок 4.28 – Выполнение тестовой ситуации №2.

Результат – списки матчей успешно формируются для различного количества команд участников соревнования.

Тестовая ситуация №3 – проверка пользователем результатов проведения матча. Результат выполнения тестовой ситуации №3 представлен на рисунке 4.29.

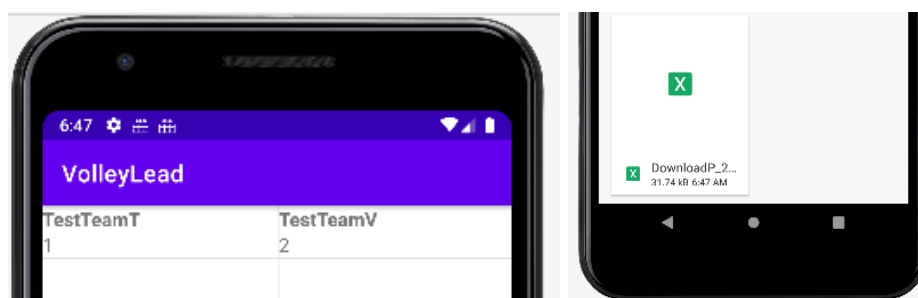


Рисунок 4.29 – Выполнение тестовой ситуации №3.

Результат – Данные о финальном счете проводимого матча заносятся в базу данных, протокол проводимого соревнования формируется, заполняется и создается в файловой системе мобильного устройства.

Тестовая ситуация №4 – проверка работоспособности веб-табло. Результат выполнения тестовой ситуации представлена на рисунке 4.30.

Партия	
4	
Раунд	
10	
Команда	Команда
TestTeamZ	TestTeamV
Счет по раундам	Счет по раундам
6	4
Счет по партиям	Счет по партиям
1	2

Рисунок 4.30 – выполнение тестовой ситуации №4.

Результат – данные успешно отправляются в базу данных выб-табло, после чего успешно отображаются в полях вывода.

Исходя из данных полученных в ходе РФТ все функциональные требования представленные в разделе 1.1 выполнены.

## **Заключение**

В рамках выпускной квалификационной работы были выполнены следующие задачи:

- произведен анализ предметной области;
- произведено изучение существующих аналогов;
- произведено определение требований к программе;
- произведен выбор инструментов разработки;
- произведен выбор архитектуры проектируемой программной системы;
- произведено моделирование логики проектируемой системы;
- разработан графический интерфейс приложений проектируемой программной системы;
- разработаны логическая и физическая модели баз данных проектируемой системы;
- разработано мобильное приложение, реализующее полный функционал описанный на этапе постановки требований к программе;
- Разработан модуль трансляции статистических данных матчей на веб-табло.

Таким образом поставленная изначально цель по разработке программной системы, переносящей процессы подготовки и проведения соревнований по волейболу в цифровую среду полностью выполнена. Разработка всех требуемых программных модулей завершена.

## Список литературы

1. А.М. Васильев, Баринов Е.И., «Программное обеспечение для организации и проведения соревнований по игровым видам спорта» [Электронный ресурс] – режим доступа: [https://conf.tusur.ru/api/v1/reports/29597/download?role=spa&file\\_type=thesis](https://conf.tusur.ru/api/v1/reports/29597/download?role=spa&file_type=thesis)
2. Официальный сайт программы «Спортивные таблицы» [Электронный ресурс] – режим доступа: <http://sport-tables.narod.ru/>
3. Официальный сайт программы «Tournament planner» [Электронный ресурс] – режим доступа: <https://www.tournamentsoftware.com/product/page.aspx?id=3&s=2>
4. Документ «положение о проведении Первенства томской области по волейболу» [Электронный ресурс] – режим доступа: <https://pandia.ru/text/77/344/60054.php>
5. Сайт «Про свободное ПО» [Электронный ресурс] – режим доступа: <https://pro-spo.ru/linuxprog/236--eclipse->
6. Сайт «Реляционные СУБД» [Электронный ресурс] – режим доступа: <https://tproger.ru/translations/sqlite-mysql-postgresql-comparison/>
7. Официальный сайт «Open Server Panel» [Электронный ресурс] – режим доступа: <https://ospanel.io/>
8. Официальный сайт «MySQL» [Электронный ресурс] – режим доступа: <https://www.mysql.com/>
9. Сайт «Архитектура Клиент-Сервер» [Электронный ресурс] – режим доступа: <https://itelon.ru/blog/arkhitektura-klient-server/>
10. Нотация IDEF0 – [Электронный ресурс] – режим доступа: <https://www.businessstudio.ru/wiki/docs/current/doku.php/ru/csdesign/bpmodeling/idef0>
11. Официальный сайт «Apache Poi» – [Электронный ресурс] – режим доступа: <https://poi.apache.org/>
12. Электронный учебник HTML и CSS – [Электронный ресурс] – режим доступа: [https://www.tct.ru/upload/elekt\\_uchebnik/HTMLCSS/index.html](https://www.tct.ru/upload/elekt_uchebnik/HTMLCSS/index.html)

13. Образовательный портал GeekBrains – [Электронный ресурс] – режим доступа: <https://gb.ru/posts/cto-takoe-css-obyasnyam-prostymi-slovami>
14. Руководство по PHP – [Электронный ресурс] – режим доступа: <https://www.php.net/manual/ru/intro-what-is.php>