Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИУ «Информатика и системы управления»

КАФЕДРА ИУ-7 «Программное обеспечение ЭВМ и информационные технологии»

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К КУРСОВОЙ РАБОТЕ

НА ТЕМУ:

«Разработка базы данных для проведения соревнований Российской Федерации Подводного Рыболовства»

Студент группы **ИУ7-64Б** _____ **П. А. Егорова**

(Подпись, дата) (И.О. Фамилия)

Руководитель _____ **Т. А. Никульшина**

(Подпись, дата) (И.О. Фамилия)

***2023** г.*

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Автоматизация — процесс, освобождающий человека от множества механических задач и помогающий ему повысить производительность труда. Автоматизация затронула большое количество процессов в современном мире: от работы кассира в магазине и подсчета стоимости покупок до вычисления значений физических величин в работе ученых. Также подсчет очков на соревнованиях различных дисциплин производится автоматически. Однако существуют организации, судейство соотязаний которых происходит вручную: Российская Федерация Подводного Рыболовства (далее: РФПР) занимается проведением соревнования по подводной охоте [1]. Деятельность федерации, связанная с подсчетом очков и ведением архива участников соревнований, не автоматизирована.

Целью курсовой работы является проектирование и разработка базы данных для проведения соревнований Российской Федерации Подводного Рыболовства.

Для достижения поставленной цели, необходимо решить следующие задачи:

- определить необходимый функционал приложения, предоставляющего доступ к базе данных;
- выделить роли пользователей приложения, а также формализовать данные;
- проанализировать системы управления базами данных и выбрать подходящую систему для хранения данных;
- спроектировать базу данных, описать ее сущности и связи;
- реализовать интерфейс для доступа к базе данных.

Итогом работы станет приложение, предоставляющее доступ к базе данных.

1 Аналитический раздел

В данном разделе будут выдвинуты требования к приложению, определены пользователи системы, формализованы хранимые о системе данные, проведен анализ существующих моделей баз данных и осуществлен ее выбор.

1.1 Требования к приложению

Чтобы спортсмены имели возможность сразу же после выхода из воды взвесить улов, взвешивание и судейство происходит прямо на берегу водоемов, где возможность найти место и поставить ноутбук предоставляется редко. Разрабатываемое приложение было бы удобно использовать на более компактных устройствах, таких как телефон или планшет. В связи с чем выбор пал на платформу iOS — создаваемое программное обеспечение будет разработано для смартфонов и электронных планшетов компании Apple.

Приложение должно поддерживать определенный функционал:

— персонализация для судей и администраторов;
— возможность входа без авторизации для спортсменов;
— формирование соревнований и их этапов;
— создание команды, деление участников по командам;
— добавление информации об улове участников;
— подсчет очков участников, команд;
— формирование личного и командного рейтингов;
— поиск определенного участника или конкретной команды.

1.2 Пользователи системы

В приложении будут представлены следующие роли:

1) Участник — пользователь, обладающий возможностью просматривать командный и личный рейтинги по любым этапам соревнований. Роль не требует авторизации.
2) Судья — пользователь, обладающий возможностями участника, а также возможностью создавать и удалять соревнования, создавать, редактиро-

вать и удалять участников, команды, добавлять участников в команды и удалять из команд, добавлять, редактировать и удалять уловы участников в этапы соревнований. Роль требует авторизации.

3) Администратор — пользователь, обладающий возможностями участника и судьи, а также возможностью редактировать и удалять профили судей. Роль требует авторизации.

На рисунке 1 представлена диаграмма использования приложения.

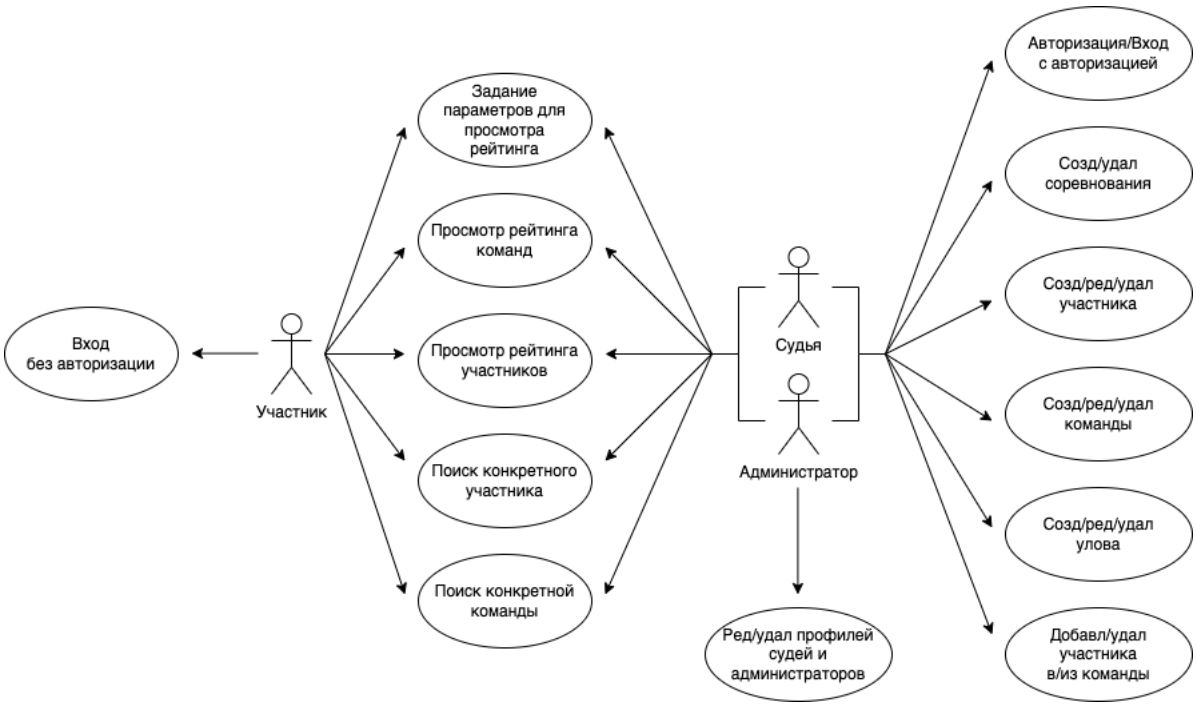

Рисунок 1 – Диаграмма использования приложения

1.3 Формализация данных

База данных должна хранить информацию о следующих сущностях

— соревнование;
— команда;
— участник;
— этап;
— название этапа;
— улов;
— рыба;

— роль;

— пользователь;

— авторизация.

В таблице 1 представлены сущности и сведения о них.

Таблица 1 – Сущности и сведения о них

Сущность	**Сведения**
Участник	Фамилия, имя, отчество, команда, город, дата рождения, очки
Команда	Название, соревнования, очки
Соревнования	Название, команды
Этап	Название этапа, участник, соревнование, очки
Название этапа	Название
Улов	Рыба, этап, вес, очки
Рыба	Название
Роль	Название
Пользователь	Роль, авторизация
Авторизация	Логин, пароль

1.4 Анализ моделей баз данных

Модель данных — это абстрактное, самодостаточное, логическое определение объектов, операторов и прочих элементов, в совокупности составляющих абстрактную машину доступа к данным, с которой взаимодействует пользователь. Эти объекты позволяют моделировать структуру данных, а операторы — поведение данных [3].

В настоящее время разработано множество моделей данных, рассмотрим основные из них.

1.4.1 Иерархическая модель

В иерархической модели данных используется представление базы данных в виде древовидной структуры, состоящей из объектов различных уровней. Между объектами существуют связи, каждый объект может включать в себя несколько объектов более низкого уровня. Такие объекты находятся в отношении предка к потомку, при этом возможна ситуация, когда объект–предок имеет несколько потомков, тогда как у объекта–потомка обязателен только один предок.

1.4.2 Сетевая модель

Сетевая модель базы данных подразумевает, что у родительского элемента может быть несколько потомков, а у дочернего элемента — несколько предков. Структура такой модели представлена в виде графа, причем каждая вершина графа хранит экземпляры сущностей (записи одного типа) и сведения о групповых отношениях с сущностями других типов. Каждая запись может хранить произвольное количество значений атрибутов (элементов данных и агрегатов), характеризующих экземпляр сущности. Для каждого типа записи выделяется первичный ключ — атрибут, значение которого позволяет однозначно идентифицировать запись среди экземпляров записей данного типа [4].

1.4.3 Реляционная модель

Реляционная модель данных является совокупностью данных и состоит из набора двумерных таблиц. При табличной организации отсутствует иерархия элементов. Таблицы состоят из строк — записей и столбцов — полей. На пересечении строк и столбцов находятся конкретные значения. Для каждого поля определяется множество его значений. За счет возможности просмотра строк и столбцов в любом порядке достигается гибкость выбора подмножества элементов.

Наиболее популярными реляционными СУБД являются Oracle, Microsoft, SQL Server и PostgreSQL.

1.5 Выбор СУБД

Система управления базами данных, сокр. СУБД — совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных [2].

Основными функциями СУБД являются:

— управление данными во внешней памяти;

— управление данными в оперативной памяти с использованием дискового кэша;

— журнализация изменений, резервное копирование и восстановление базы данных после сбоев;

— поддержка языков БД.

Для ускорения быстродействия разрабатываемого приложения, можно прибегнуть к кэшированию данных. Для кэширования данных можно использовать NoSQL [5] in–memory базы данных. Такие базы данных хранят данные в оперативной памяти, что обеспечивает более быстрый доступ к данным.

Основными решениями интеграции баз данных в мобильные приложения под iOS являются SQLite, Firestore и Realm.

1.5.1 SQLite

SQLite — это встроенная в процесс библиотека, которая реализует автономный, бессерверный транзакционный компонент SQL [6]. Код для SQLite находится в общественном достоянии и, таким образом, свободен для использования в любых целях. SQLite — это встроенный движок базы данных SQL.

В отличие от большинства других баз данных SQL, SQLite не имеет отдельного серверного процесса. Транзакции являются ACID, даже если они прерваны системными сбоями или сбоями питания. SQLite

1.5.2 Firestore

Cloud Firestore — это NoSQL, ориентированная на документы база данных. В отличие от базы данных SQL, здесь нет таблиц или строк. Вместо этого

вы храните данные в документах, которые организованы в коллекции [7].

Каждый документ содержит набор пар ключ–значение. Облачное хранилище Firestore оптимизировано для хранения больших коллекций небольших документов.

Все документы должны храниться в коллекциях. Документы могут содержать вложенные коллекции и вложенные объекты, оба из которых могут включать примитивные поля, такие как строки, или сложные объекты, такие как списки.

Коллекции и документы создаются неявно в Cloud Firestore. Просто назначьте данные документу в коллекции. Если коллекция или документ не существуют, Cloud Firestore создает их.

1.5.3 Realm от MongoDB

Realm — это реактивная, объектно-ориентированная, кроссплатформенная, NoSQL мобильная база данных [8].

Файлы Realm содержат объектные данные со следующими структурами данных: группы, таблицы, деревья кластеров и кластеры. База данных Realm организует эти структуры данных в древовидную структуру следующего вида:

- верхний уровень, известный как Группа, хранит метаданные объекта, журнал транзакций и коллекцию таблиц;
- каждый класс в схеме realm соответствует таблице в группе верхнего уровня;
- каждый таблица содержит дерево кластеров, реализацию дерева B +;
- листья на дереве кластеров называются кластерами, каждая содержит диапазон объектов, отсортированных по значению ключа;
- кластеры хранят объекты в виде наборов столбцов.

База данных Realm гарантирует, что транзакции совместимы с ACID, а также имеет удобный SDK для внедрения в мобильные приложения.

1.5.4 Выбор СУБД

NoSQL базы данных набирают популярность, причиной чему является простота разработки, функционала и высокая производительности. В связи с наличием интереса к данному подходу проектирования систем, а также из–за наличия удобного SDK для iOS, был выбран Realm от MongoDB.

1.6 Вывод из раздела

В данном разделе были выделены ролевые модели системы, конкретизированы данные и их связь между собой, построены соответствующие диаграммы. Был проведен анализ и выбор модели данных, а также СУБД, подходящей для решения поставленной задачи.

2 Конструкторский раздел

В данном разделе будут представлены этапы проектирования базы данных, выделены конкретные действия ролевой модели, спроектирован триггер и описаны основы проектирования приложения.

2.1 Проектирование базы данных

На основе выделенных ранее сущностей спроектированы следующие объекты базы данных.

1) Participant — содержит информацию об участнике и имеет следующие поля:
   - participantId — уникальный идентификатор участника;
   - participantLastname — фамилия участника;
   - participantFirstname — имя участника;
   - participantPatronymic — отчество участника;
   - participantTeam — команда участника;
   - participantCity — город проживания участника;
   - participantBirthday — дата рождения участника;
   - participantScore — очки участника.
2) Team — содержит информацию о команде и имеет следующие поля:
   - teamId — уникальный идентификатор команды;
   - teamName — название команды;
   - teamCompetitions — соревнования, в которых принимает участие команда;
   - teamScore — очки команды.
3) Competition — содержит информацию о соревновании и имеет следующие поля:
   - competitionId — уникальный идентификатор соревнования;
   - competitionName — название соревнования;
   - competitionTeams — команды, участвующие в соревнованиях.

4) Step — содержит информацию об этапе соревнования для участника:
   - stepId — уникальный идентификатор этапа;
   - stepName — название этапа;
   - stepParticipant — участник, которому принадлежит этап;
   - stepCompetition — соревнование, которому принадлежит этап;
   - lootScore — очки за зачет.
5) StepName — содержит информацию о названии этапа и имеет следующие поля:
   - stepNameId — уникальный идентификатор названия этапа;
   - stepName — название этапа.
6) Loot — содержит информацию об улове участника:
   - lootId — уникальный идентификатор улова;
   - lootFish — название рыбы;
   - lootWeight — вес рыб;
   - lootStep — зачет, к которому относится улов;
   - lootScore — очки за рыбу.
7) Fish — содержит информацию о рыбе и имеет следующие поля:
   - fishId — уникальный идентификатор рыбы;
   - fishName — название рыбы.
8) Role — содержит информацию о роли и имеет следующие поля:
   - roleId — уникальный идентификатор роли;
   - roleName — название роли.
9) User — содержит информацию о пользователе и имеет следующие поля:
   - userId — уникальный идентификатор пользователя;
   - userAuthorization — авторизация пользователя;
   - userRole — роль пользователя.
10) Authorization — содержит информацию об авторизации и имеет следующие поля:
   - authorizationId — уникальный идентификатор авторизации;

— authorizationLogin — логин;

— authorizationPassword — пароль.

Соответствующая диаграмма по описанным выше данным представлена на рисунке 2 .

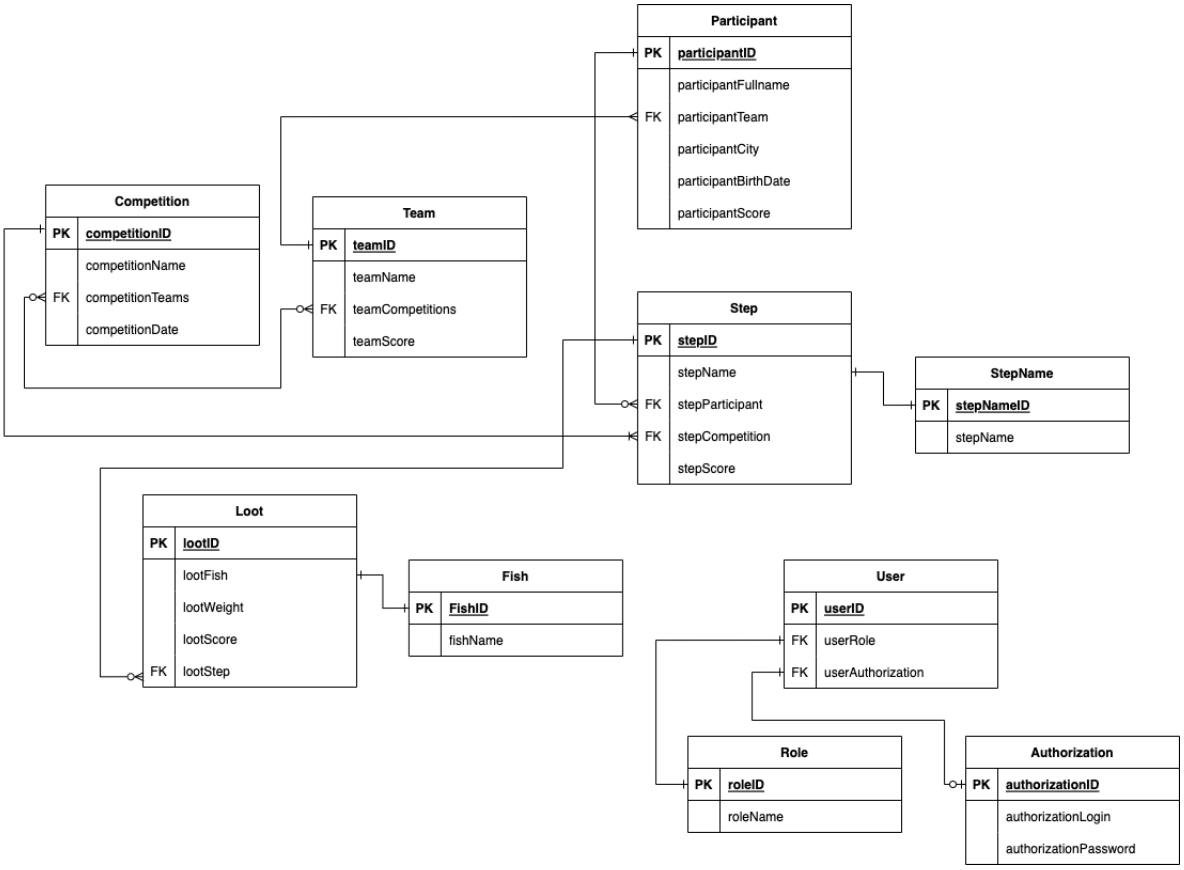

Рисунок 2 – ER–диаграмма

2.2 Ролевая модель

Ролевая модель предполагает наличие трех ролей: участника, судьи и администратора. Стоит отметить, что судья обладает всеми правами участника, а администратор — всеми правами судьи.

Для успешной реализации поставленной задачи программа должна предоставлять следующие возможности:

— просмотр профилей участников;
— просмотр профилей команд;
— просмотр личного и командного рейтингов;
— просмотр личного и командного рейтингов по этапам;
— сортировка личного и командного рейтингов по убыванию/по возрастанию;
— сортировка личного и командного рейтингов по этапам по убыванию/по возрастанию.

Дополнительные возможности судьи:

— регистрация;
— авторизация;
— создание/удаление соревнования;
— создание/редактирование/удаление команды;
— создание/редактирование/удаление участника;
— создание/редактирование/удаление улова;
— добавление улова в этап;
— добавление участника в команду;
— добавление команды в соревнование.

Дополнительные возможности администратора:

— редактирование/удаление профилей судей;
— создание/редактирование/удаление профилей администраторов.

2.3 Триггер

Триггер — это хранимая процедура особого типа, которую пользователь не вызывает непосредственно, а исполнение которой обусловлено действием по модификации данных: добавлением, модификацией, удалением строки в заданной таблице.

В каждой из сущностей, таких как улов, этап, участник и команда, представлено поле «очки». Когда рыба взвешена и очки за нее необходимо добавить в этап участника, присвоить участнику и его команде, нужно в каждой сущности, связанной с уловом, обновить значение поля.

Связи соответствующих сущностей представлены на рисунке 3.

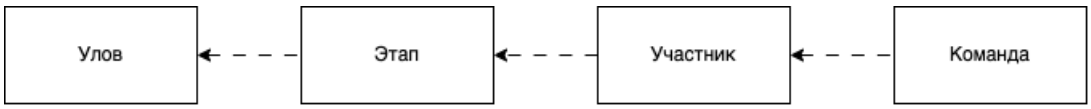

Рисунок 3 – Диаграмма связей сущностей при обновлении очков

Для процедуры перерасчета очков было бы удобно создать триггер, который автоматически обновляет значение соответствующего поля в описанных четырех сущностях, при добавлении, редактировании и удалении улова.

Схема алгоритма триггера представлена на рисунке 3.

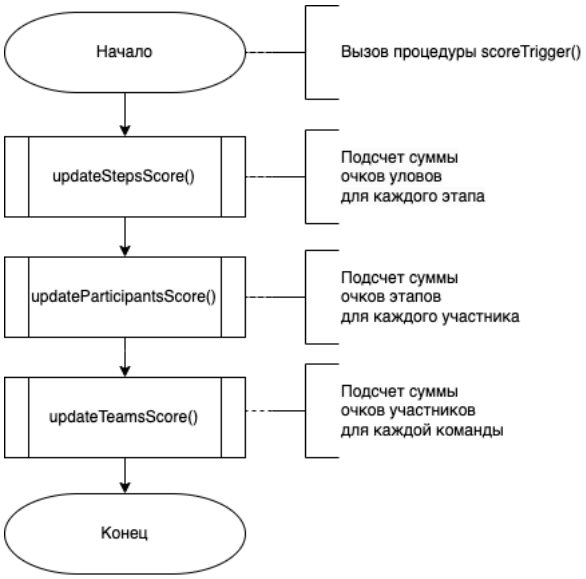

Рисунок 4 – Триггер для обновления очков

2.4 Проектирование приложения

При разработке приложение для iOS может быть разделено на 3 этапа в соответствии с принципами «чистой архитектуры» [9]:

1) слой доступа к данным — подключение к базе данных, отправка запросов, получение информации из базы данных;
2) слой пользовательского интерфейса (UI) — взаимодействие пользователя с программой;
3) слой бизнес логики — взаимодействия между слоем доступа к данным и приложением.

2.5 Вывод из раздела

В данном разделе были спроектированы сущности базы данных и их связи, ролевая модель и триггер, были формализованы требования к программе, описаны этапы разработки приложения.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. РФПР | Российская Федерация Подводного Рыболовства [Электронный ресурс]. — Режим доступа: http://rfpr.su/, свободный (дата обращения: 16.04.2023)

2. Что такое СУБД — RU—CENTER [Электронный ресурс]. — Режим доступа: https://www.nic.ru/help/chto-takoe-subd_8580.html, свободный (дата обращения: 16.04.2023)

3. Дейт К.Дж. Введение в системы баз данных. — 8–е изд. — М.: «Вильямс», 2006.

4. Сетевая СУБД [Электронный ресурс]. — Режим доступа: https://www.tadviser.ru/index.php/Статья:Сетевая_СУБД, свободный (дата обращения: 16.04.2023)

5. Что такое NoSQL? | Amazon AWS [Электронный ресурс]. — Режим доступа: https://aws.amazon.com/ru/nosql, свободный (дата обращения: 16.04.2023)

6. About SQLite [Электронный ресурс]. — Режим доступа: https://sqlite.org/about.html, свободный (дата обращения: 16.04.2023)

7. Документация | Firebase Documentation [Электронный ресурс]. — Режим доступа: https://firebase.google.com/docs?hl=ru, свободный (дата обращения: 16.04.2023)

8. Realm Database [Электронный ресурс]. — Режим доступа: https://www.mongodb.com/docs/realm/sdk/swift/realm-database/, свободный (дата обращения: 16.04.2023)

9. Мартин Р. Чистая архитектура. Искусство разработки программного обеспечения. – СПб.: Питер, 2018. – 352 с.