

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ
Федеральное государственное бюджетное образовательное учреждение высшего
образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет
«Вычислительная техника»
(наименование)

Кафедра
«Системы автоматизированного проектирования»
(наименование)

Направление подготовки

09.03.01 «Информатика и вычислительная техника»
(код и наименование)

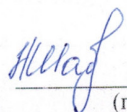
Профиль

«Системы автоматизированного проектирования»
(наименование)

БАКАЛАВРСКАЯ РАБОТА
на тему

Разработка информационной системы
судейства и хронометража по плаванию

Студент

 29.05.17
(подпись, дата)

Шабанов Н.В.
(фамилия, инициалы)

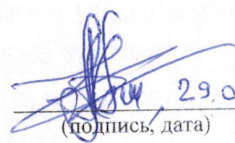
Руководитель

 29.05.17
(подпись, дата)

Эль В.В.
(фамилия, инициалы)

Нормоконтролёр

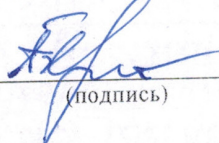
информ. каф. САПР ПГУ
(должность, место работы)

 29.05.17
(подпись, дата)

Бондарь Д.С.
(фамилия, инициалы)

Работа допущена к защите (протокол заседания кафедры от _____ № _____)

Заведующий кафедрой


(подпись)

Варшавский А.М.
(фамилия, инициалы)

Работа защищена с отметкой _____ (протокол заседания ГЭК от _____ № _____)

Секретарь ГЭК

_____ (подпись)

_____ (фамилия, инициалы)

Пенза 2017

СОДЕРЖАНИЕ

Введение	5
1 Анализ технического задания	7
1.1 Анализ предметной области	7
1.2 Обзор аналогов	9
1.3 Обоснование использования средств разработки	14
2 Проектирование базы данных	17
2.2 Концептуальное проектирование	17
2.3 Создание даталогической модели данных	18
2.4 Создание физической модели данных	20
2.5 Описание процесса создания базы данных	22
3 Разработка пользовательского приложения	23
3.1 Разработка пользовательского интерфейса	23
3.2 Разработка запросов к базе данных	29
4 Тестирование	32
Заключение	37
Список использованной литературы	38
Приложение А	40
Приложение Б	43

Введение

Сегодня информационные технологии нашли довольно широкое применение в сфере спорта. Сегодня невозможно представить себе организацию крупных спортивных соревнований без использования современных информационных технологий [1].

В циклических видах спорта (бег, академическая гребля, плавание) информационные технологии в основном используются для точной фиксации результата. Используется множество различных современных технологий спортивного хронометража, такие как: инфракрасные створы, стартовые калитки, контактные ленты, контактные финишные панели, фотофиниш и др. Также активно используется программное обеспечение, позволяющее вести учет спортсменов, участвующих в соревнованиях. Как правило оно из себя представляет базу данных и приложение, позволяющее работать с ней.

Целью выпускной квалификационной работы является создание базы данных и разработка приложения, обеспечивающее хранение, обработку данных о соревнованиях и спортсменах, участвующих в них.

Задачи выпускной квалификационной работы:

1. Анализ предметной области
2. Выполнение сравнительного анализа аналогов
3. Проектирование и создание базы данных
4. Разработка приложения
5. Тестирование разработанного приложения

Одной из задач данной работы является сравнительный анализ систем судейства и хронометража для проведения соревнований по плаванию. К сожалению, отечественные производители по прежнему не могут конкурировать с зарубежными аналогами, что доказывается использованием швейцарской системы Swiss Timing на универсиаде 2013 и чемпионате мира по водным видам спорта 2015 в Казани.

Из этого следует, что разработка отечественной системы судейства и хронометража для плавания вопрос актуальный и требующий ско-

рейшего разрешения.

Кроме того, на сегодняшний день рынок систем судейства и хронометража по плаванию представлен малым количеством таких систем, что говорит его перспективности.

ГЛАВА 1. АНАЛИЗ ТЕХНИЧЕСКОГО ЗАДАНИЯ

В данной главе будут рассмотрены системы автоматического хронометража для проведения соревнований по плаванию, произведен их сравнительный анализ. Также будут рассмотрены средства, с помощью которых будет разрабатываться заявленная информационная система.

1.1. Анализ предметной области

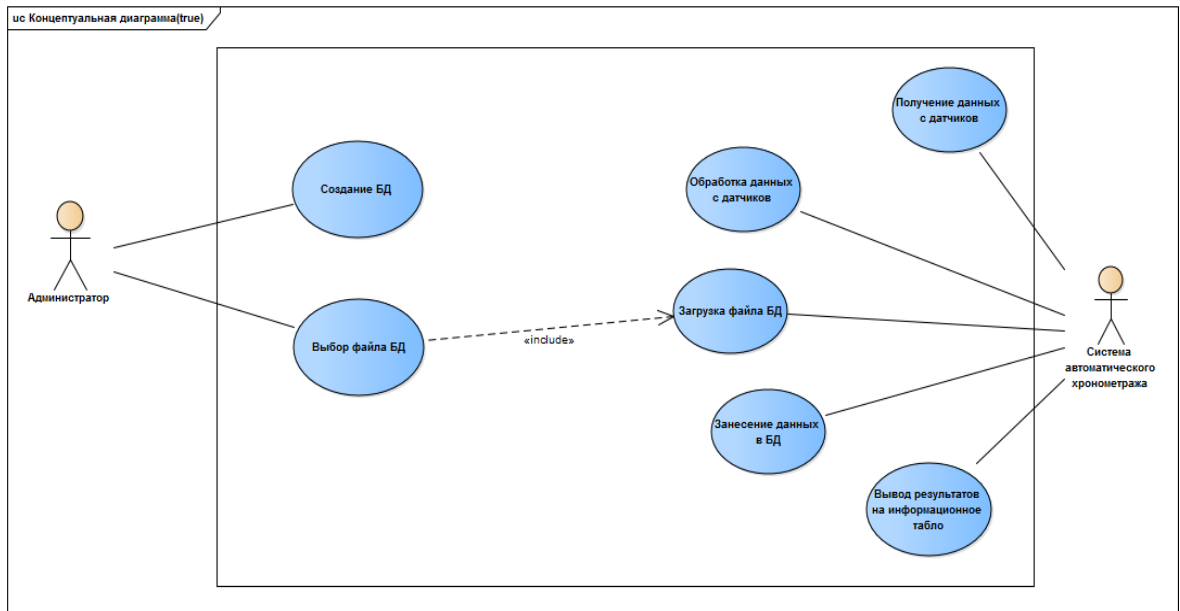
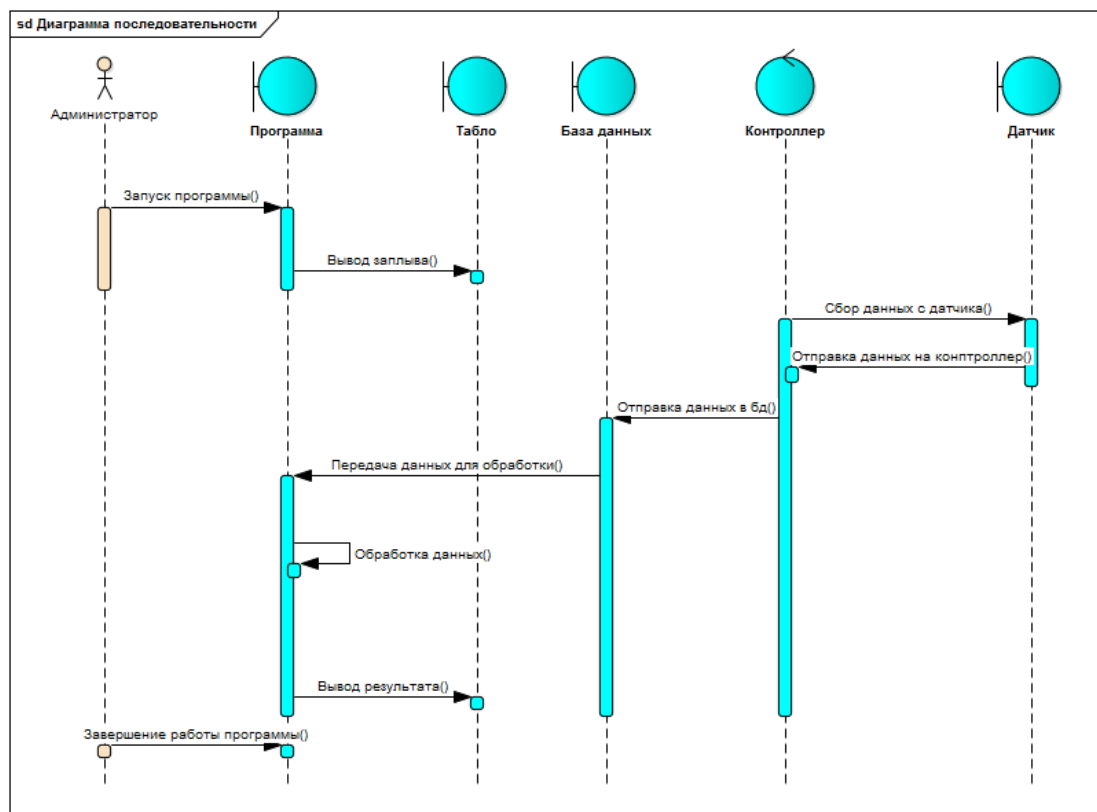
Люди всегда любили соревноваться. Во все века они выясняют кто быстрее, кто сильнее и выносливее. Для объективного выявления сильнейших всегда нужны три вещи:

1. правила или регламент, по которому проводятся состязания;
2. равные условия для всех участников;
3. независимая и не предвзятая третья сторона — спортивный судья, который фиксирует результаты состязания [2].

Пытаясь устранить недостатки ручного измерения, повысить точность измерений независимо от количества спортсменов, люди активно используют современные технологии. Другими словами, сейчас наблюдается тенденция вытеснения человеческого фактора в сфере судейства соревнований. Так как плавание является циклическим видом спорта, то и при проведении соревнований по плаванию используются системы судейства и хронометража. Главной целью такой системы является определение времени прохождения дистанции каждым из участников.

В состав современной системы судейства и хронометража по плаванию как правило входят следующие компоненты:

1. программное обеспечение;
2. контроллер-хронометр;
3. информационное табло;
4. терминал связи;
5. устройство подачи стартового сигнала;
6. оконечные устройства (финишные панели – TouchPad, кнопки судей-секундометристов).



1.2 Обзор аналогов

В качестве объектов сравнения были отобраны три системы, представленные на российском рынке и одна на зарубежном.

1. "БТ-40", ООО "Счетные системы";
2. "PT-7000", ООО "Виктори";
3. "Swim Mini/Midl/Max", ООО "Альт";
4. Swiss Timing, компания OMEGA.

Целевыми пользователями рассматриваемых систем судейства и хронометража являются физкультурно-спортивные организации, которые предоставляют имеющиеся в наличии спортивные объекты для проведения соревнований различного уровня.

Для проведения сравнительного анализа был сформирован перечень критериев, описывающих основные характеристики систем судейства и хронометража, рассматриваемые заказчиком на этапе выбора системы:

- ☐ функциональность;
- ☐ производительность и масштабируемость;
- ☐ интегрируемость;
- ☐ ценовая политика;
- ☐ соответствие официальным требованиям спортивных федераций;
- ☐ сервисная поддержка.

1.2.1 Система " БТ-40"

Автоматические и полуавтоматические системы регистрации результатов для плавания на базе контроллера БТ-40 «Нерпа» [3] предназначены для проведения заплывов во всех видах плавания.

Данный контроллер принимает данные с терминалов находящихся на каждой дорожке. Количество обслуживаемых дорожек зависит от выбранной конфигурации системы и уровня проводимых соревнований.

Судья-секундометрист на каждой дорожке фиксирует результат пловца с помощью гарнитуры "Стоп", подключенной к терминалу, при этом подается сиг-

нал сирены. Старт дает судья-стартер с помощью гарнитуры "Старт", подключаемой к контроллеру.

Результаты автоматически сортируются, отображаются на табло и передаются на компьютер. Управление системой возможно полностью с компьютера.

Позволяет осуществлять формирование стартового и итогового протоколов соревнований с автоматическим подсчетом полученных разрядов и очков с последующим выводом протоколов на печать.

Данная система предназначена для проведения соревнований республиканского масштаба.

Функциональные возможности системы включают в себя:

- ☐ секундомер с точностью до 0,01 секунды для каждой дорожки;
- ☐ подача звукового сигнала настраиваемой длительности и тональности отдельно по каждой дорожке;
- ☐ автоматическая сортировка результатов на табло;
- ☐ отображение данных пловца для каждой дорожки;
- ☐ отображение вида плавания и другой информации о заплыве;
- ☐ отображение любых сообщений на «бегущей строке»;
- ☐ предусмотрено отключение с компьютера части табло, чтобы в дни, свободные от соревнований, отображать только текущее время, температуру и влажность.

Кроме того осуществляется интеграция с персональным компьютером и принтером для сохранения результатов соревнований в базе данных и распечатки стартовых и итоговых протоколов соревнований.

Система легко масштабируется и позволяет изменять конфигурацию:

- ☐ увеличивать число обслуживаемых дорожек до 8-ми;
- ☐ полностью автоматизировать систему, добавив в нее финишные и фальстартовые панели;
- ☐ заменить используемое информационное табло.

Стоимость системы колеблется в зависимости от количества обслуживаемых дорожек и степени автоматизации системы: от 901 000 рублей за полуавто-

матическую систему на 3 дорожки до 2 060 000 рублей за комплектацию на 8 дорожек с финишными панелями шириной 2,4 метра [3].

1.2.2 Система "Pt-7000"

По данным официального сайта ООО «Виктори», основными преимуществами системы судейства и хронометража PT-7000 [4] являются:

- ☐ автоматическое резервное копирование данных для минимизации ошибок при эксплуатации;
- ☐ все функции, необходимые хронометрической системе для плавания включены в одно устройство для обеспечения стабильной и постоянной работы;
- ☐ разработана специально для плавания – нет необходимости в многофункциональных компьютерах или ПО;
- ☐ специальные клавиши для выбора спортивного события, диодный дисплей для подтверждения выбранных настроек упрощает использование;
- ☐ соответствует требованиям ФИНА;
- ☐ сохраняет в памяти результаты 300 соревнований;
- ☐ удобочитаемый усиленный ЖК дисплей;

Конфигурация и комплектность данной системы подбирается с учетом бюджета и требований заказчика. Так же производитель предоставляет полную сервисную поддержку на протяжении всего гарантийного периода.

Система судейства и хронометража от ООО «Виктори» предоставляет возможность выбора длины бассейна (25 или 50 метров), а также возможность выбора дистанции (от 25 метров и более). Максимальное время измерения – 59 минут 59 секунд.

Кроме того, система масштабируема, что проявляется в количестве обслуживаемых дорожек (числом до 10) и добавлении финишных и фальстартовых панелей, и обладает возможностью интеграции с ПК, ноутбуком и различными моделями информационных табло.

Имеется возможность реализации беспроводного управления системой.

Официальный сайт ООО «Виктори» [4] не предоставляет открытой информации о стоимости своих систем и услуг по их установке ввиду того, что подбор конфигурации для каждого заказчика индивидуален.

1.2.3 Система "Swim Mini/Midl/Max"

ООО «Альт» предоставляет на выбор несколько конфигураций систем судейства и хронометража Swim [5]:

Mini – это самый простой вариант построения системы хронометража для 25 метрового бассейна на 6 дорожек. Стартовый бокс по нажатию стартовой кнопки, обнуляет время системного таймера и формирует мощный звуковой сигнал, по которому спортсмены стартуют. Финишный бокс регистрирует нажатие кнопок, и передает это событие системному таймеру. Таймер считает время с момента старта до нажатия кнопки финиша для каждой из шести дорожек, отправляет эти данные на принтер чеков, информационное табло, и на компьютер. Существует так же версия Mini 6T, в комплект которой входит информационное табло.

Вариант Midl отличается от Mini возможностью обслуживать 8 дорожек 50 метрового бассейна в полуавтоматическом режиме.

Конфигурация под названием Max имеет два варианта обслуживания восьми дорожек 50 метрового бассейна в автоматическом режиме, комплектуясь восьмью и шестнадцатью контактными финишными панелями.

Каждая из конфигураций способна обслуживать соревнования с печатью информационных чеков и выводом информации на табло и компьютер. Также каждая конфигурация позволяет фиксировать результаты в полуавтоматическом режиме. Но, только конфигурация Mini 6T имеет в комплекте поставки информационное табло.

Стоит отметить так же, что масштабируемость у самых дешевых систем, при приобретении готовой системы, отсутствует и доукомплектовать их, в случае необходимости, не удастся. Масштабируемость же более богатых конфигураций остается на уровне и выбор числа обслуживаемых дорожек, а также докупка и подключение дополнительного оборудования не составит проблем.

Информации о соответствии требованиям спортивных федераций официальный сайт не предоставляет.

Цены на приобретение готовых систем варьируются от 198 000 рублей за самую дешевую конфигурацию до 2 580 000 рублей за самую дорогую. При этом стоит отметить, что для большинства конфигураций информационное табло в комплектацию не входит, а самое дешевое табло на сайте ООО «Альт» имеет стоимость 220 000 рублей [5].

1.1.4 Система «Swiss Timing»

Производителем данной системы является швейцарская компания «OMEGA». На протяжении десятилетий система судейства и хронометража Swiss Timing [5] является официальным партнером Международного олимпийского комитета, и имеет богатый опыт проведения Олимпийских игр и чемпионатов мира по разным видам спорта. Система Swiss Timing соответствуют правилам проведения соревнований и отвечают всем требованиям технических комиссий федераций.

Как заявляет сайт производителя, данная система удобна в использовании, защищена от неверных действий оператора, выполняет все необходимые операции для проведения соревнований, соответствует последним изменениям в правилах соревнований, не содержит ошибок. Программное обеспечение — это, в первую очередь, практический опыт разработчиков компании «OMEGA» в проведении соревнований, регистрации рекордов, создании систем хронометража разных уровней и разных комплектаций.

Официальным представителем компании «OMEGA» в России является компания «Спорт Лайн».

1.1.5 Сравнение аналогов

Изучив аналоги, можно сделать сравнительный анализ относительно ранее обозначенных критериев. Результаты сравнительного анализа представлены в табл.1.

Таблица 1. Результаты сравнительного анализа аналогов

Параметр	Система "БТ-40"	Система "РТ-7000"	Система "Swim Mini/Midl/Max"	Система «Swiss Timing»
Информативность сайта системы	4	3	3	4
Масштабируемость	5	4	3	5
Соответствие требованиям спортивных федераций	4	5	4	5
Интегрируемость	4	4	5	5
Сервисная поддержка	Есть	Есть	н/д	Есть
Стоимость, руб	2 060 000	н/д	2 800 000	3 200 000

Обозначения:

н/д – нет данных

Из таблицы видно, что действительно, зарубежный аналог по нескольким показателям обходит наши отечественные аналоги. Однако и стоимость у данной системы выше. Также следует отметить, что программное обеспечение только системы БТ-40 обеспечивает автоматическое формирование стартовых и итоговых протоколов, но данная система не предназначена для проведения международных соревнований и не может быть масштабируема до 10-ти дорожек.

1.3 Обоснование использования средств разработки

В рамках выпускной квалификационной работы было принято решение разработать информационную систему, включающую базу данных соревнований, спортсменов и т.д. и приложение по ее сопровождению.

Поэтому для разработки информационной системы судейства и хронометража по плаванию была выбрана среда Microsoft Visual Studio 2015 и язык программирования C#.

Microsoft Visual Studio — линейка продуктов компании Майкрософт, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств. Данные продукты позволяют разрабатывать как консольные приложения, так и приложения с графическим интерфейсом.

Visual Studio включает в себя редактор исходного кода с поддержкой технологии IntelliSense и возможностью простейшего рефакторинга кода. Встроенный отладчик может работать как отладчик уровня исходного кода, так и как отладчик машинного уровня. Остальные встраиваемые инструменты включают в себя редактор форм для упрощения создания графического интерфейса приложения, веб-редактор, дизайнер классов и дизайнер схемы базы данных. VisualStudio позволяет создавать и подключать сторонние дополнения (плагины) для расширения функциональности практически на каждом уровне, включая добавление поддержки систем контроля версий исходного кода (например, Subversion и VisualSourceSafe), добавление новых наборов инструментов (например, для редактирования и визуального проектирования кода).

Главным преимуществом Visual Studio 2015 является производительность. Обеспечивает возможность создания разнообразных приложений на основе одного набора навыков.

C# — объектно-ориентированный язык программирования. C# относится к семье языков с C-подобным синтаксисом, из них его синтаксис наиболее близок к C++ и Java.

Язык имеет статическую типизацию, поддерживает полиморфизм, перегрузку операторов (в том числе операторов явного и неявного приведения типа), делегаты, атрибуты, события, свойства, обобщённые типы и методы, итераторы, анонимные функции с поддержкой замыканий, LINQ, исключения, комментарии в формате XML [7].

В главе 1 был проведен анализ предметной области, на основе которого были определены компоненты, входящие в состав системы судейства и хронометража по плаванию. Были представлены UML диаграммы, описывающие взаимодействие этих компонентов между собой и с пользователем. Далее был выбран перечень аналогов на отечественном и зарубежном рынке. Изучив выбранные аналоги,

был проведен сравнительный анализ, который показал, что отечественные разработки, действительно, не могут конкурировать с рассмотренным зарубежным аналогом. Далее были выбраны средства, с помощью которых будет производиться проектирование базы данных и приложения, обеспечивающего работу с этой базой данных. В этих целях была выбрана и дана краткая характеристика среде Microsoft Visual Studio 2015 Community и языку программирования C#.

ГЛАВА 2. ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ

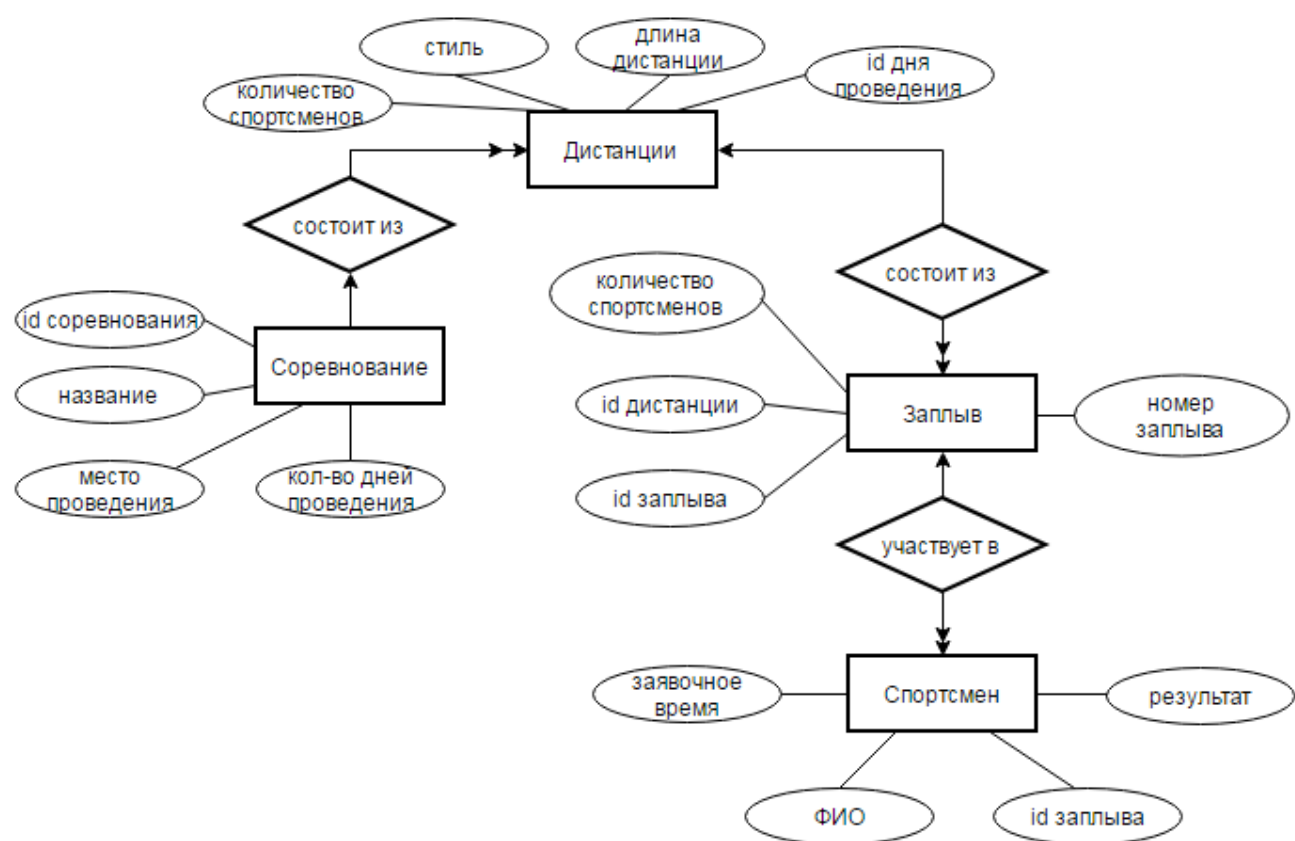
2.1 Концептуальное проектирование

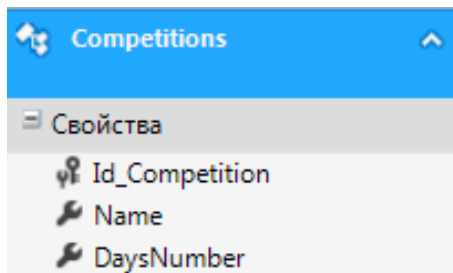
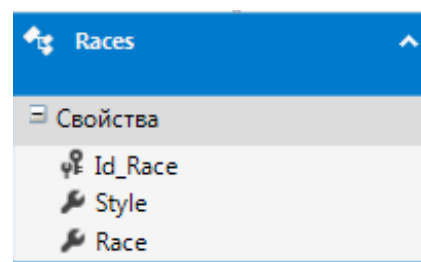
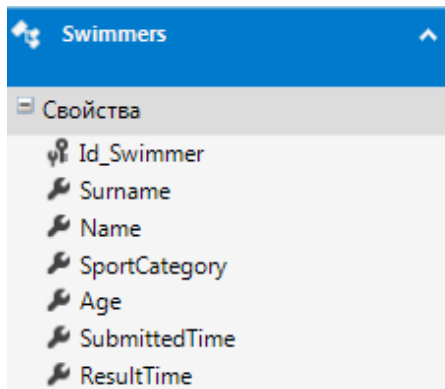
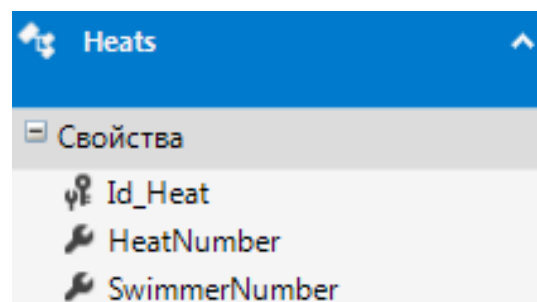
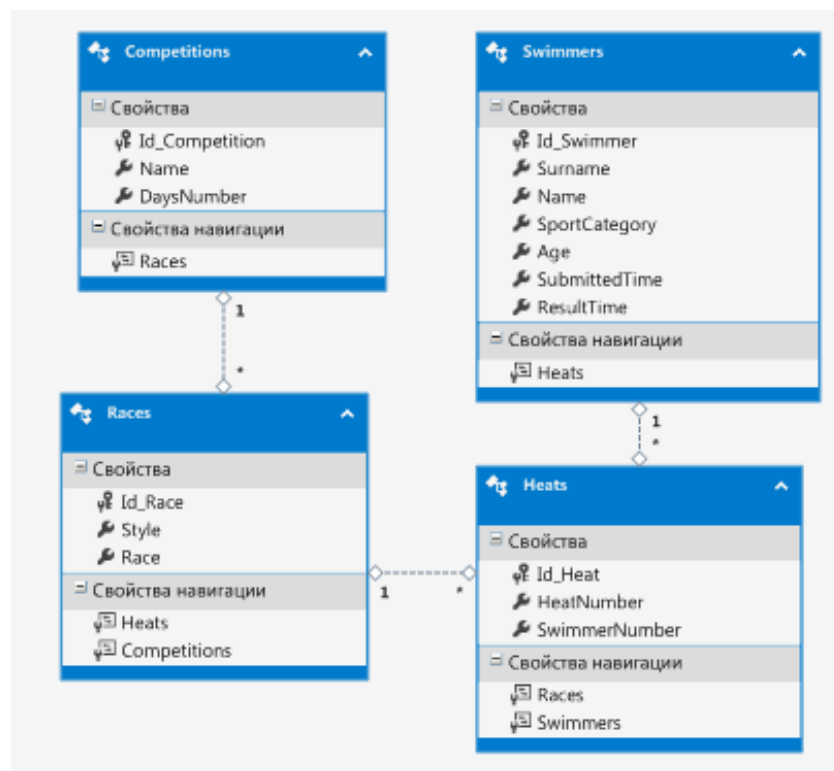
Концептуальная модель - это отражение предметной области, для которой разрабатывается база данных. Такая модель создаётся без ориентации на какую-либо конкретную СУБД и модель данных. Для ее проектирования потребуется выделить сущности, их атрибуты и тип связи между сущностями [8]. Выделенные сущности и атрибуты представлены в табл. 2.

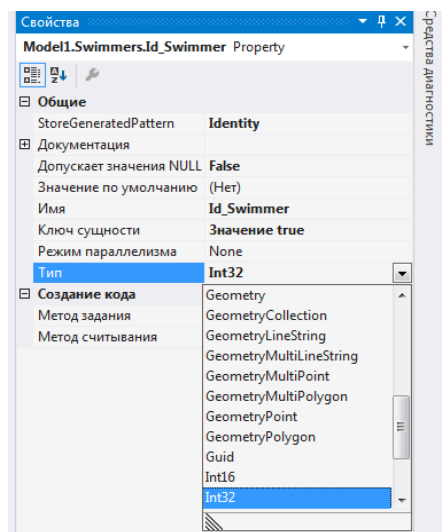
Таблица 2 - Выделенные сущности и атрибуты

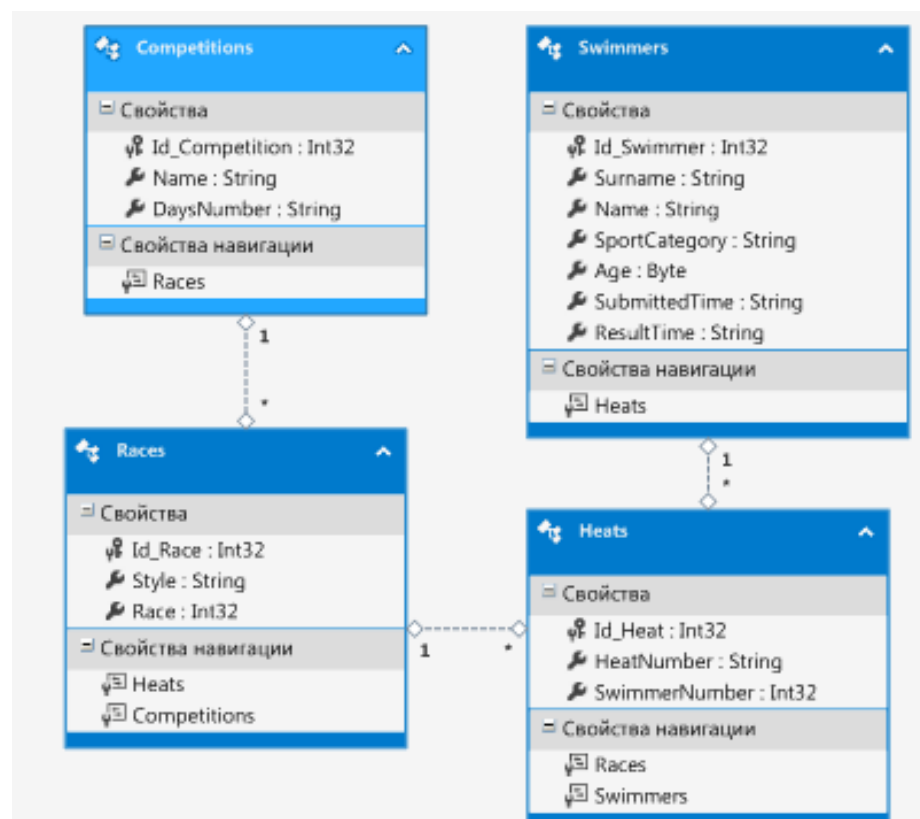
Таблица	Атрибут
Соревнование	ID соревнования
	Название
	Место проведения
	Количество дней
Дистанции	ID Дистанции
	Длина дистанции
	Стиль плавания
	Количество спортсменов
Заплыв	ID Дистанции
	ID Заплыва
	Номер заплыва
	Количество спортсменов
Спортсмен	ID Заплыва
	ФИО пловца
	Заявочное время
	Результат

Далее нужно продумать, какими отношениями будут связаны выделенные сущности. Для наглядности представления следует спроектировать концептуаль-









Подключиться

Журнал

Обзор

Type here to filter the list

Локально

Сеть

Azure

Имя сервера:

(localdb)\MSSQLLocalDB

Проверка подлинности:

Проверка подлинности Windows

Имя пользователя:

Николай-ПК\Николай

Пароль:

☐ Запомнить пароль

Имя базы данных:

<по умолчанию>

Дополнительно...

Подключиться

Отмена

ван код классов на языке программирования C#, на котором будет осуществляться взаимодействие с разработанной базой данных.

ГЛАВА 3. РАЗРАБОТКА ПОЛЬЗОВАТЕЛЬСКОГО ПРИЛОЖЕНИЯ

В данной главе будет описана разработка приложения, позволяющего работать с созданной базой данных. Для этого будут использованы возможности Entity Framework [10], чтобы реализовать методы чтения, удаления, изменения и вставки данных – методы CRUD (create read update delete).

Для разработки приложения была выбрана Windows Presentation Foundation (WPF) - система для построения клиентских приложений Windows с визуально привлекательными возможностями взаимодействия с пользователем, графическая (презентационная) подсистема в составе .NET Framework (начиная с версии 3.0), использующая язык XAML. В основе WPF лежит векторная система визуализации, не зависящая от разрешения устройства вывода и созданная с учётом возможностей современного графического оборудования. WPF предоставляет средства для создания визуального интерфейса, включая язык XAML (eXtensible Application Markup Language), элементы управления, привязку данных, макеты, двухмерную и трёхмерную графику, анимацию, стили, шаблоны, документы, текст, мультимедиа и оформление. Графической технологией, лежащей в основе WPF, является DirectX, в отличие от Windows Forms, где используется GDI/GDI+. Производительность WPF выше, чем у GDI+ за счёт использования аппаратного ускорения графики через DirectX.

3.1 Разработка пользовательского интерфейса

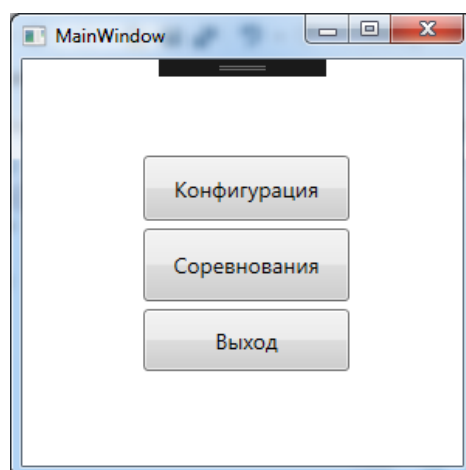
Интерфейс - некоторый способ (стандарт) взаимодействия (обмена информацией, данными) между программой и человеком, другой программой и т.п [11]. Под графическим пользовательским интерфейсом (GUI - Graphical User Interface) понимается некоторая система (среда), служащая для интерфейса прикладных программ с пользователем на основе многооконного представления данных. Прикладные программные интерфейсы (API -Application program interface) выполняют программно-языковые функции для связи пользовательских приложений с GUI. Программист может специфицировать функции (окна, меню, процессы отображения, пиктограммы), которые необходимы для соответствующих при-

кладных программ. API включает средства, используемые разработчиками программ при создании GUI для конкретных приложений. В любой прикладной программе почти половина всего кода программы служит именно для организации интерфейса - ввод/вывод информации, работа с мышью, организация меню, реакция на ошибки и т. п.

Эффективность интерфейса заключается в быстром, насколько возможно, развитии у пользователей простой концептуальной модели взаимодействия. Концепция согласованности интерфейса состоит в том, что при работе с компьютером у пользователя формируется система ожидания одинаковых реакций на одинаковые действия, что постоянно подкрепляет пользовательскую модель интерфейса. Другой составляющей интерфейса является свойство его конкретности и наглядности. Оно обеспечивается применением в панелях различных цветов и других выразительных средств. Идеи и концепции затем обретают физическое отображение на экране, с которым непосредственно взаимодействует пользователь. Интерфейс может быть согласован в трех аспектах или категориях: физической, синтаксической и семантической.

- ☐ Физическая согласованность относится к техническим средствам;
- ☐ синтаксическая согласованность относится к последовательности и порядку появления элементов на экране (язык представления) и последовательности запросов (язык действий);
- ☐ семантическая согласованность относится к значению элементов, составляющих интерфейс.

Согласованный интерфейс дает пользователям и разработчикам экономию времени и средств. Пользователям понадобится меньше времени, чтобы научиться использовать приложения и для выполнения работы. Согласованный интерфейс сокращает число ошибок пользователя и способствует тому, что пользователь чувствует себя с системой комфортнее. Согласованный пользовательский интерфейс выгоден и разработчикам приложений, он позволяет выделить общие блоки интерфейса, стандартизировать элементы интерфейса и взаимодействие с ними.



Выбор соревнования

Name	DaysNumber
Первенство области	2
Чемпионат России	3
Первенство ПФО	3

Название соревнования

Количество дней

Добавить

RaceHeatWindow

ХРОНОМЕТРАЖ

ДИСТАНЦИИ

Style	Race
Баттерфляй	100
Спина	200
Вольный	800

ЗАПЛЫВЫ

СОСТАВ УЧАСТНИКОВ СОРЕВНОВАНИЯ

Name	Surname	SportCategory	Age
Иван	Иванов	МС	20
Александр	Курынов	МС	20
Миша	Боровков	Любитель	19

WindowAddRace

Стиль

Длина дистанции

Добавить

WindowAddSwimmer

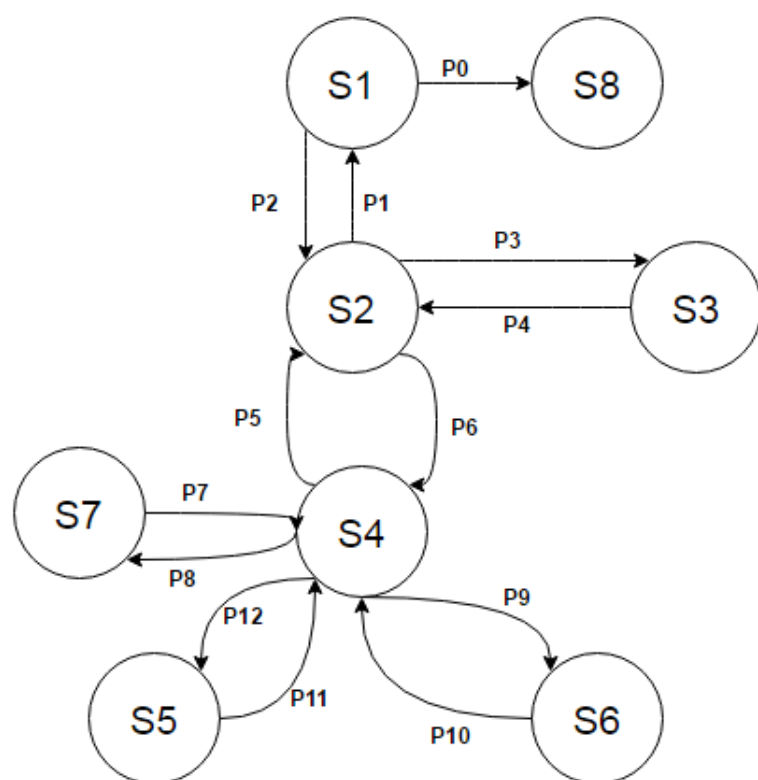
Фамилия

Имя

Возраст

Категория

Добавить



P6 – нажатие кнопки «Загрузить соревнование»

P7 – нажатие кнопки «Закрыть»

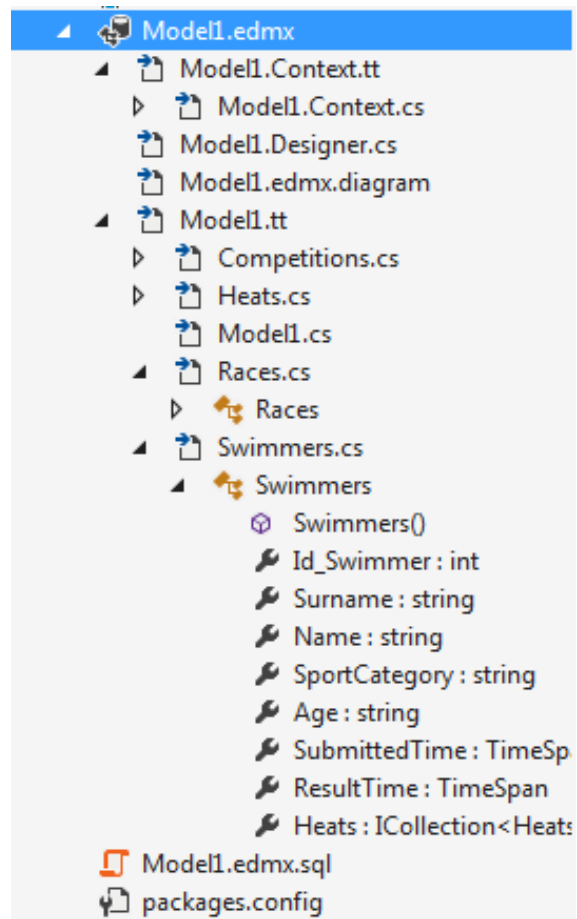
P8 - нажатие кнопки «Сформировать результаты»

P9 – нажатие кнопки «Добавить»

P10 - нажатие кнопки «Закрыть»

P11 - нажатие кнопки «Закрыть»

P12 – нажатие кнопки «Добавить»



```
dbContext.CompetitionsSet.Add(new Competitions
{
    Name = textBox.Text,
    DaysNumber = textBox1.Text,
});
dbContext.SaveChanges();
LoadedCompetition();
```

```
Competitions competition = dbContext.CompetitionsSet
    .Where(c => c.DaysNumber == "2")
    .FirstOrDefault();
```

```
dbContext.CompetitionsSet.Remove(competition);
dbContext.SaveChanges();
```

```
IQueryable<Races> races = dbContext.RacesSet;
var queryRace =
    from race in races
    select new
    {
        race.Style,
        race.Race
    };
dataGridViewRaces.ItemsSource = queryRace.ToList();
```

```
var competition = dbContext.CompetitionsSet
    .Where(c => c.DaysNumber == "2")
    .FirstOrDefault();
// Внести изменения
competition.DaysNumber = "3";
dbContext.SaveChanges();
```

ГЛАВА 4. ТЕСТИРОВАНИЕ

Тестирование программного обеспечения (software testing) — деятельность, выполняемая для оценки и улучшения качества программного обеспечения [12]. Эта деятельность, в общем случае, базируется на обнаружении дефектов и проблем в программных системах. Тестирование программных систем состоит из динамической верификации поведения программ на конечном (ограниченном) наборе тестов (set of test cases), выбранных соответствующим образом из обычно выполняемых действий прикладной области и обеспечивающих проверку соответствия ожидаемому поведению системы.

Цели тестирования ПО:

1. повысить вероятность того, что приложение, предназначенное для тестирования, будет работать правильно при любых обстоятельствах;
2. повысить вероятность того, что приложение, предназначенное для тестирования, будет соответствовать всем описанным требованиям;
3. предоставление актуальной информации о состоянии продукта на данный момент.

В рамках выпускной квалификационной работы было выбрано функциональное тестирование - это тестирование ПО в целях проверки реализуемости функциональных требований, то есть способности ПО в определённых условиях решать задачи, нужные пользователям. Функциональные требования определяют, что именно делает ПО, какие задачи оно решает [13].

Согласно техническому заданию приложение должно выполнять следующие функции:

1. ввод данных о соревнованиях, дистанциях, спортсменов;
2. редактирование данных о спортсменах;
3. удаление данных о соревнованиях, дистанциях, спортсменах, заплывах;
4. выполнять формирование стартовых протоколов;
5. выполнять формирование итоговых протоколов с результатами.

Далее будут рассмотрены тесты на каждую вышеуказанную функцию.

Выбор соревнования

Name	DaysNumber	
Первенство области	2	
Чемпионат России	3	
Первенство ПФО	3	
Первенство Европы	4	

Название соревнования

Первенство Европы

Количество дней

4

Добавить

Открыть



СОСТАВ УЧАСТНИКОВ СОРЕВНОВАНИЯ

Name	Surname	SportCategory	Age	
Александр	Курынов	МС	20	
Алексей	Боровков	Любитель	19	

			Максимальное количество строк: 1000		
	Id_Swimmer	Surname	Name	SportCategory	Age
	1	Курынов	Александр	МС	20
	2	Боровков	Алексей	МС	19



СОСТАВ УЧАСТНИКОВ СОРЕВНОВАНИЯ

Name	Surname	SportCategory	Age
Миша	Боровков	Любитель	19
Иван	Иванов	МС	20
Александр	Курынов	МС	20



СОСТАВ УЧАСТНИКОВ СОРЕВНОВАНИЯ

Name	Surname	SportCategory	Age
Александр	Курынов	МС	20
Миша	Боровков	Любитель	19

3. Ожидаемый результат – в открывшемся окне отобразится список спортсменов в зависимости от занятого ими места;
4. Полученный результат – в открывшемся окне отобразился список спортсменов в зависимости от занятого ими места.

Таким образом, в главе 4 было дано определение понятию тестирование, определены цели тестирования. На основе полученных знаний было проведено функциональное тестирование разработанного приложения, которое показало, что приложение выполняет все функции, заявленные в техническом задании.

ЗАКЛЮЧЕНИЕ

В рамках данной выпускной квалификационной работы был проведен анализ предметной области, на основе которого были определены компоненты, входящие в состав системы судейства и хронометража по плаванию. Были представлены UML диаграммы, описывающие взаимодействие этих компонентов между собой и с пользователем. Далее был выбран перечень аналогов на отечественном и зарубежном рынке. Изучив выбранные аналоги, был проведен сравнительный анализ, который показал, что отечественные разработки, действительно, не могут конкурировать с рассмотренным зарубежным аналогом. Для разработки базы данных и приложения, обеспечивающего работу с этой базой данных была выбрана и дана краткая характеристика среде Microsoft Visual Studio 2015 Community и языку программирования C#

Далее были рассмотрены 3 основных этапа проектирования базы данных: концептуальное проектирование, даталогическое проектирование, физическое проектирование. На каждом этапе в итоге была получена соответствующая схема модели данных. На основе физической модели был сгенерирован скрипт на языке запросов T-SQL, из которого в свою очередь получены классы на языке программирования C#, с помощью которых можно осуществлять запросы к базе данных.

Также были рассмотрены основные принципы разработки пользовательского интерфейса. В результате этого был разработан дружелюбный интерфейс, который был описан при помощи графа диалога. После этого, благодаря использованию Entity Framework, были написаны запросы к базе данных на языке программирования C#, что существенно сократило трудоемкость данного этапа разработки.

После разработки приложения было проведено функциональное тестирование, которое показало, что приложение выполняет все функции, заявленные в техническом задании.

Целевыми пользователями разработанной информационной системы судейства и хронометража являются спортивные объекты, используемые для проведения соревнований различного уровня.

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. Трощенко О.Н. Использование информационных технологий при проведении спортивных мероприятий/ Трощенко О.Н., Шлемова М.В., Чернышева И.В.// [Текст]. М.: «ЭЛОС» – 2013 – с. 191-192
2. Анализ информационных технологий в области физической культуры и спорта – Cyberleninka [Электронный ресурс]. <http://cyberleninka.ru/article/n/analiz-informatsionnyh-tehnologiy-v-oblasti-fizicheskoy-kultury-i-sporta>, свободный, (дата обращения: 8.02.2017);
3. Система регистрации результатов БТ-40 – ООО «Счетные системы» [Электронный ресурс]. <http://www.sc-sys.ru/sport/swimming/timing/bt40>, свободный, (дата обращения: 8.02.2017);
4. Хронометраж для плавания РТ-7000 – Спортивные табло VICTORY [Электронный ресурс]. <http://victorytablo.ru/novosti1/hronometrazh-dlya-plavaniya-pt-7000.html> , свободный, (дата обращения: 8.02.2017);
5. Обзор конфигураций систем судейства и хронометража Swim от ООО «Альт» [Электронный ресурс]. <http://altazar.ru/gr6000.html>, свободный, (дата обращения: 8.02.2017).
6. Об оборудовании Swiss Timing – Сведения о компании «OMEGA». [Электронный ресурс]. https://www.sport-line.ru/timing/about_ST/, свободный (дата обращения: 1.03.2017).
7. Фленов М. Библия C#. – 3-е изд. [Текст]. М.: «БХВ-Петербург» — 2004— с. 1 - 87с.
8. Кириллов В. Введение в реляционные базы данных. [Текст]. М.: «БХВ-Петербург», 2013 – с. 67 – 78.
9. Разработка приложения Entity Framework 6 - Использование Model-First [Электронный ресурс]. https://professorweb.ru/my/entity-framework/6/level1/1_5.php, свободный (дата обращения: 6.04.2017).

10. Model First – Описание разработки базы данных. [Электронный ресурс]. <https://metanit.com/sharp/entityframework/2.5.php>, свободный (дата обращения: 20.04.2017).
11. Тидвелл Д. Разработка пользовательских интерфейсов. [Текст]. Пер. с англ. - М.: "Питер", 2014 – с. 101 – 110.
12. Тамре Л. Введение в тестирование программного обеспечения. [Текст]. Пер. с англ. - М.: Издательский дом "Вильямс", 2013 – с. 15 – 30
13. Блэк Р. Ключевые процессы тестирования: Планирование, подготовка, проведение, совершенствование. [Текст]. Пер. с англ. - М.: "Лори", 2014 – с. 27 – 72.

ПРИЛОЖЕНИЕ А

Листинг SQL-скрипта

```
SET QUOTED_IDENTIFIER OFF;
GO
USE [swim];
GO
IF SCHEMA_ID(N'dbo') IS NULL EXECUTE(N'CREATE SCHEMA [dbo]');
GO

-- -----
-- Внешние ключи
-- -----

IF OBJECT_ID(N'[dbo].[FK_RacesHeats]', 'F') IS NOT NULL
    ALTER TABLE [dbo].[HeatsSet] DROP CONSTRAINT [FK_RacesHeats];
GO
IF OBJECT_ID(N'[dbo].[FK_SwimmersHeats]', 'F') IS NOT NULL
    ALTER TABLE [dbo].[HeatsSet] DROP CONSTRAINT [FK_SwimmersHeats];
GO
IF OBJECT_ID(N'[dbo].[FK_CompetitionsRaces]', 'F') IS NOT NULL
    ALTER TABLE [dbo].[RacesSet] DROP CONSTRAINT [FK_CompetitionsRaces];
GO

-- -----
-- Таблицы
-- -----

IF OBJECT_ID(N'[dbo].[RacesSet]', 'U') IS NOT NULL
    DROP TABLE [dbo].[RacesSet];
GO
IF OBJECT_ID(N'[dbo].[SwimmersSet]', 'U') IS NOT NULL
    DROP TABLE [dbo].[SwimmersSet];
GO
IF OBJECT_ID(N'[dbo].[HeatsSet]', 'U') IS NOT NULL
    DROP TABLE [dbo].[HeatsSet];
GO
IF OBJECT_ID(N'[dbo].[CompetitionsSet]', 'U') IS NOT NULL
    DROP TABLE [dbo].[CompetitionsSet];
GO

-- -----
-- Создание полей таблиц
-- -----

-- Создание таблицы 'RacesSet'
CREATE TABLE [dbo].[RacesSet] (
    [Id_Race] int IDENTITY(1,1) NOT NULL,
    [Style] nvarchar(max) NOT NULL,
    [Race] int NOT NULL,
    [Competitions_Id_Competition] int NOT NULL
);
GO

-- Создание таблицы 'SwimmersSet'
CREATE TABLE [dbo].[SwimmersSet] (
    [Id_Swimmer] int IDENTITY(1,1) NOT NULL,
    [Surname] nvarchar(max) NOT NULL,
    [Name] nvarchar(max) NOT NULL,
    [SportCategory] nvarchar(max) NOT NULL,
    [Age] tinyint NOT NULL,
    [SubmittedTime] nvarchar(max) NULL,
    [ResultTime] nvarchar(max) NULL
);
GO
```

```

-- Создание таблицы 'HeatsSet'
CREATE TABLE [dbo].[HeatsSet] (
    [Id_Heat] int IDENTITY(1,1) NOT NULL,
    [HeatNumber] nvarchar(max) NOT NULL,
    [SwimmerNumber] int NULL,
    [Races_Id_Race] int NOT NULL,
    [Swimmers_Id_Swimmer] int NOT NULL
);
GO

-- Создание таблицы 'CompetitionsSet'
CREATE TABLE [dbo].[CompetitionsSet] (
    [Id_Competition] int IDENTITY(1,1) NOT NULL,
    [Name] nvarchar(max) NOT NULL,
    [DaysNumber] nvarchar(max) NOT NULL
);
GO

-- -----
-- Создание первичных ключей
-- -----

-- Создание первичного ключа в таблице 'RacesSet'
ALTER TABLE [dbo].[RacesSet]
ADD CONSTRAINT [PK_RacesSet]
    PRIMARY KEY CLUSTERED ([Id_Race] ASC);
GO

-- Создание первичного ключа в таблице 'SwimmersSet'
ALTER TABLE [dbo].[SwimmersSet]
ADD CONSTRAINT [PK_SwimmersSet]
    PRIMARY KEY CLUSTERED ([Id_Swimmer] ASC);
GO

-- Создание первичного ключа в таблице 'HeatsSet'
ALTER TABLE [dbo].[HeatsSet]
ADD CONSTRAINT [PK_HeatsSet]
    PRIMARY KEY CLUSTERED ([Id_Heat] ASC);
GO

-- Создание первичного ключа в таблице 'CompetitionsSet'
ALTER TABLE [dbo].[CompetitionsSet]
ADD CONSTRAINT [PK_CompetitionsSet]
    PRIMARY KEY CLUSTERED ([Id_Competition] ASC);
GO

-- -----
-- Creating all FOREIGN KEY constraints
-- -----

-- Creating foreign key on [Races_Id_Race] in table 'HeatsSet'
ALTER TABLE [dbo].[HeatsSet]
ADD CONSTRAINT [FK_RacesHeats]
    FOREIGN KEY ([Races_Id_Race])
    REFERENCES [dbo].[RacesSet]
        ([Id_Race])
    ON DELETE NO ACTION ON UPDATE NO ACTION;
GO

-- Creating non-clustered index for FOREIGN KEY 'FK_RacesHeats'
CREATE INDEX [IX_FK_RacesHeats]
ON [dbo].[HeatsSet]
    ([Races_Id_Race]);
GO

```

```

-- Creating foreign key on [Swimmers_Id_Swimmer] in table 'HeatsSet'
ALTER TABLE [dbo].[HeatsSet]
ADD CONSTRAINT [FK_SwimmersHeats]
    FOREIGN KEY ([Swimmers_Id_Swimmer])
    REFERENCES [dbo].[SwimmersSet]
        ([Id_Swimmer])
    ON DELETE NO ACTION ON UPDATE NO ACTION;
GO

-- Creating non-clustered index for FOREIGN KEY 'FK_SwimmersHeats'
CREATE INDEX [IX_FK_SwimmersHeats]
ON [dbo].[HeatsSet]
    ([Swimmers_Id_Swimmer]);
GO

-- Creating foreign key on [Competitions_Id_Competition] in table 'RacesSet'
ALTER TABLE [dbo].[RacesSet]
ADD CONSTRAINT [FK_CompetitionsRaces]
    FOREIGN KEY ([Competitions_Id_Competition])
    REFERENCES [dbo].[CompetitionsSet]
        ([Id_Competition])
    ON DELETE NO ACTION ON UPDATE NO ACTION;
GO

-- Creating non-clustered index for FOREIGN KEY 'FK_CompetitionsRaces'
CREATE INDEX [IX_FK_CompetitionsRaces]
ON [dbo].[RacesSet]
    ([Competitions_Id_Competition]);
GO

```

ПРИЛОЖЕНИЕ Б

Листинг программы

Файл Model.tt

```
namespace Testing
{
    using System;
    using System.Collections.Generic;
    // класс сущности «Соревнования»
    public partial class Competitions
    {
        [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage",
"CA2214:DoNotCallOverridableMethodsInConstructors")]
        public Competitions()
        {
            this.Races = new HashSet<Races>();
        }

        public int Id_Competition { get; set; }
        public string Name { get; set; }
        public string DaysNumber { get; set; }

        [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage",
"CA2227:CollectionPropertiesShouldBeReadOnly")]
        public virtual ICollection<Races> Races { get; set; }
    }
    // класс сущности «Заплывы»
    public partial class Heats
    {
        public int Id_Heat { get; set; }
        public string HeatNumber { get; set; }
        public Nullable<int> SwimmerNumber { get; set; }

        public virtual Races Races { get; set; }
        public virtual Swimmers Swimmers { get; set; }
    }
    // класс сущности «Дистанции»
    public partial class Races
    {
        [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage",
"CA2214:DoNotCallOverridableMethodsInConstructors")]
        public Races()
        {
            this.Heats = new HashSet<Heats>();
        }

        public int Id_Race { get; set; }
        public string Style { get; set; }
        public int Race { get; set; }

        [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage",
"CA2227:CollectionPropertiesShouldBeReadOnly")]
        public virtual ICollection<Heats> Heats { get; set; }
        public virtual Competitions Competitions { get; set; }
    }
    // класс сущности «Спортсмены»
    public partial class Swimmers
    {
        [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage",
"CA2214:DoNotCallOverridableMethodsInConstructors")]
        public Swimmers()
        {
            this.Heats = new HashSet<Heats>();
        }
    }
}
```

```

    }

    public int Id_Swimmer { get; set; }
    public string Surname { get; set; }
    public string Name { get; set; }
    public string SportCategory { get; set; }
    public byte Age { get; set; }
    public string SubmittedTime { get; set; }
    public string ResultTime { get; set; }

    [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage",
"CA2227:CollectionPropertiesShouldBeReadOnly")]
    public virtual ICollection<Heats> Heats { get; set; }
}

// класс-контейнер базы данных
public partial class Model1Container : DbContext
{
    public Model1Container()
        : base("name=Model1Container")
    {
    }

    protected override void OnModelCreating(DbModelBuilder modelBuilder)
    {
        throw new UnintentionalCodeFirstException();
    }

    public virtual DbSet<Races> RacesSet { get; set; }
    public virtual DbSet<Swimmers> SwimmersSet { get; set; }
    public virtual DbSet<Heats> HeatsSet { get; set; }
    public virtual DbSet<Competitions> CompetitionsSet { get; set; }
}

```

Файл RaceHeatWindow.xaml.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;

namespace Testing
{
    /// <summary>
    /// Логика взаимодействия для RaceHeatWindow.xaml
    /// </summary>
    public partial class RaceHeatWindow : Window
    {
        public RaceHeatWindow()
        {
            InitializeComponent();
        }
        Model1Container dbContext = new Model1Container();

        private void LoadSwimmers(object sender, RoutedEventArgs e)
        {
            IQueryable<Swimmers> swimmers = dbContext.SwimmersSet;

```

```

        var querySwimmers =
            from swimmer in swimmers
            select new
            {
                swimmer.Name,
                swimmer.Surname,
                swimmer.SportCategory,
                swimmer.Age,
                swimmer.SubmittedTime
            };
        dataGridSwimmer.ItemsSource = querySwimmers.ToList();
    }

    private void LoadRace(object sender, RoutedEventArgs e)
    {
        IQueryable<Races> races = dbContext.RacesSet;
        var queryRace =
            from race in races
            select new
            {
                race.Style,
                race.Race
            };
        dataGridRaces.ItemsSource = queryRace.ToList();
    }

    private void AddSwimmer(object sender, RoutedEventArgs e)
    {
        WindowAddSwimmer windowAddSwimmer = new WindowAddSwimmer();
        windowAddSwimmer.Show();
        LoadSwimmers(sender, e);
    }

    private void AddRace(object sender, RoutedEventArgs e)
    {
        WindowAddRace windowAddRace = new WindowAddRace();
        windowAddRace.Show();
    }
}
}

```

Файл WindowAddSwimmer.xaml.cs

```

namespace Testing
{
    /// <summary>
    /// Логика взаимодействия для WindowAddSwimmer.xaml
    /// </summary>
    public partial class WindowAddSwimmer : Window
    {
        public WindowAddSwimmer()
        {
            InitializeComponent();
        }
        Model1Container dbContext = new Model1Container();

        // Заполнение полей БД путем чтения из формы
        private void Add(object sender, RoutedEventArgs e)
        {
            try
            {
                dbContext.SwimmersSet.Add(new Swimmers
                {
                    Name = textBoxName.Text,
                    Surname = textBoxSurname.Text,
                    SportCategory = textBoxCategory.Text,

```

```

        Age = Convert.ToByte(textBoxAge.Text),
        SubmittedTime = textBoxSubmittedTime.Text
    });
    dbContext.SaveChanges();
}

catch (Exception)
{
    MessageBox.Show("Проверьте правильность заполнения полей");
}
}
}
}
}

```

Файл RaceHeatWindow.xaml

```

<Window x:Class="Testing.RaceHeatWindow"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    xmlns:local="clr-namespace:Testing"
    mc:Ignorable="d"
    Title="RaceHeatWindow" Height="545.963" Width="1147.827"
    Loaded="LoadSwimmers">
    <Grid>
        <DataGrid x:Name="dataGridRaces" HorizontalAlignment="Left" Margin="737,29,0,0" VerticalAlignment="Top" Height="173" Width="393"/>
        <Label x:Name="label" Content="ДИСТАНЦИИ" HorizontalAlignment="Left" Margin="850,-2,0,0" VerticalAlignment="Top" Height="44" Width="146" FontSize="20" FontWeight="Bold" FontFamily="Trebuchet MS"/>
        <Label x:Name="label1" Content="ЗАПЛЫВЫ" HorizontalAlignment="Left" Margin="850,207,0,0" VerticalAlignment="Top" Height="32" Width="146" FontSize="20" FontWeight="Bold"/>
        <DataGrid x:Name="dataGridHeats" HorizontalAlignment="Left" Margin="737,244,0,0" VerticalAlignment="Top" Height="262" Width="393"/>
        <DataGrid x:Name="dataGridSwimmer" HorizontalAlignment="Left" Margin="10,330,0,0" VerticalAlignment="Top" Height="176" Width="698"/>
        <Label x:Name="label2" Content="СОСТАВ УЧАСТНИКОВ СОРЕВНОВАНИЯ" HorizontalAlignment="Left" Margin="145,293,0,0" VerticalAlignment="Top" Height="37" Width="406" FontSize="20" FontWeight="Bold"/>
        <DataGrid x:Name="dataGridTime" HorizontalAlignment="Left" Margin="10,29,0,0" VerticalAlignment="Top" Height="264" Width="698"/>
        <Label x:Name="label3" Content="ХРОНОМЕТРАЖ" HorizontalAlignment="Left" Margin="206,-7,0,0" VerticalAlignment="Top" Height="39" Width="174" FontSize="20" FontWeight="Bold"/>
        <Button x:Name="button" Content="" HorizontalAlignment="Left" Margin="737,0,0,0" VerticalAlignment="Top" Width="34" Height="29" Click="AddRace">
            <Button.Background>
                <ImageBrush ImageSource="C:\Users\Николай\Desktop\ButtonAdd.png" />
            </Button.Background>
        </Button>
        <Button x:Name="button1" Content="" HorizontalAlignment="Left" Margin="10,291,0,0" VerticalAlignment="Top" Width="52" Click="AddSwimmer" Height="37">
            <Button.Background>
                <ImageBrush ImageSource="C:\Users\Николай\Desktop\ButtonAdd.png" />
            </Button.Background>
        </Button>
        <Button x:Name="button2" Content="" HorizontalAlignment="Left" Margin="77,290,0,0" VerticalAlignment="Top" Width="44" Height="37">
            <Button.Background>
                <ImageBrush ImageSource="C:\Users\Николай\Desktop\Remove.png" />
            </Button.Background>
        </Button>
        <Button x:Name="button3" Content="Сформировать" HorizontalAlignment="Left" Margin="737,217,0,0" VerticalAlignment="Top" Width="99"/>
    </Grid>

```

```

        <Button x:Name="button4" Content="Результаты" HorizontalAlignment="Left" Margin="10,4,0,0" VerticalAlignment="Top" Width="75"/>
    </Grid>
</Window>

```

Файл HeatWindow.xaml.cs

```

// добавление заплыва
private void button1_Click(object sender, EventArgs e)
{
    HeatForm plForm = new HeatForm();
    DialogResult result = plForm.ShowDialog(this);

    if (result == DialogResult.Cancel)
        return;

    Heats heat = new Heats ();
    Heat.Number = (int)plForm.numericUpDown1.Value;
    Heat.SwimmerNumber = plForm.textBox1.Text;
    Heat.Race = plForm.comboBox1.SelectedItem.ToString();

    db.Heat.Add(heat);
    db.SaveChanges();

    MessageBox.Show("Новый заплыв добавлен");
}

// редактирование
private void button2_Click(object sender, EventArgs e)
{
    if(dataGridView1.SelectedRows.Count>0)
    {
        int index = dataGridView1.SelectedRows[0].Index;
        int id=0;
        bool converted = Int32.TryParse(dataGridView1[0, index].Value.ToString(),out id);
        if(converted==false)
            return;

        Heats heat = db.Players.Find(id);

        HeatsForm plForm = new HeatsForm();

        plForm.numericUpDown1.Value = heat.Number;
        plForm.comboBox1.SelectedItem = heat.SwimmerNumber;
        plForm.textBox1.Text = heat.Race;

        DialogResult result = plForm.ShowDialog(this);

        if (result == DialogResult.Cancel)
            return;

        heat.Number = (int)plForm.numericUpDown1.Value;
        heat.SwimmerNumber = plForm.textBox1.Text;
        heat.Race = plForm.comboBox1.SelectedItem.ToString();

        db.SaveChanges();
        dataGridView1.Refresh(); // обновляем грид
        MessageBox.Show("Заплыв обновлен");
    }
}

// удаление
private void button3_Click(object sender, EventArgs e)
{
    if (dataGridView1.SelectedRows.Count > 0)

```

```

        {
            int index = dataGridView1.SelectedRows[0].Index;
            int id = 0;
            bool converted = Int32.TryParse(dataGridView1[0, index].Value.ToString(), out
id);
            if (converted == false)
                return;

            Heats heat = db.Heats.Find(id);
            db.Heats.Remove(heat);
            db.SaveChanges();

            MessageBox.Show("Заплыв удален");
        }
    }
}

```

Файл CompetitionWindow.xaml.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;

namespace Testing
{
    /// <summary>
    /// Логика взаимодействия для EditWindow.xaml
    /// </summary>
    public partial class CompetitionWindow : Window
    {
        public CompetitionWindow()
        {
            InitializeComponent();
        }

        Model1Container dbContext = new Model1Container();

        private void LoadedCompetition()
        {
            IQueryable<Competitions> competitions = dbContext.CompetitionsSet;
            var query =
            from competition in competitions
            select new { competition.Name,
                        competition.DaysNumber };
            dataGrid.ItemsSource = query.ToList();
        }

        private void LoadedCompetition(object sender, RoutedEventArgs e)
        {
            IQueryable<Competitions> competitions = dbContext.CompetitionsSet;

            //
            var query =
            from competition in competitions

```

```

        select new { competition.Name,
                    competition.DaysNumber };
        dataGrid.ItemsSource = query.ToList();

        var competition = dbContext.CompetitionsSet
            .Where(c => c.DaysNumber == "2")
            .FirstOrDefault();
        Внести изменения
        competition.DaysNumber = "3";
        dbContext.SaveChanges();

        ObjectQuery<Product> products = dataEntities.Products;

    private void Click_Add(object sender, RoutedEventArgs e)
    {
        dbContext.CompetitionsSet.Add(new Competitions
        {
            Name = textBox.Text,
            DaysNumber = textBox1.Text,
        });
        dbContext.SaveChanges();
        LoadedCompetition();
    }

    private void Open(object sender, RoutedEventArgs e)
    {
        RaceHeatWindow raceHeatWindow = new RaceHeatWindow();
        raceHeatWindow.Show();
        this.Close();
    }

    private void Remove(object sender, RoutedEventArgs e)
    {
        // Удаление выбранного соревнования
        Competitions competition = dbContext.CompetitionsSet
            .Where(c => c.DaysNumber == ComboBox)
            .FirstOrDefault();

        dbContext.CompetitionsSet.Remove(competition);
        dbContext.SaveChanges();
    }
}
}

```

Файл WindowAddRace.xaml.cs

```

namespace Testing
{
    /// <summary>
    /// Логика взаимодействия для WindowAddRace.xaml
    /// </summary>
    public partial class WindowAddRace : Window
    {
        public WindowAddRace()
        {
            InitializeComponent();
        }
        Model1Container dbContext = new Model1Container();

        private void AddRace(object sender, RoutedEventArgs e)
        {

```

```

        int Id = dbContext.CompetitionsSet
        .Where(c => c.Id_Competition == 1)
        .Select(c => c.Id_Competition)
        .FirstOrDefault();

        // Заполнение полей БД путем чтения из формы
        try
        {
            Races race = new Races
            {
                Style = Convert.ToString(comboBoxStyle.SelectedItem),
                Race = Convert.ToInt32(comboBoxRace.SelectedItem),
                // Competitions = Id,
            };

            dbContext.SaveChanges();
        }
        catch (Exception)
        {
            MessageBox.Show("Проверьте правильность заполнения полей");
        }
    }
}

```