Министерство образования и науки Республики Башкортостан

Государственное автономное профессиональное образовательное учреждение

Уфимский колледж статистики, информатики и вычислительной техники

|  |  |
| --- | --- |
|  | УТВЕРЖДАЮ  Заместитель директора  по учебной работе  З.З. Курмашева  « » 2024 г. |

ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ

И РАЗРАБОТКА ПРИЛОЖЕНИЯ ДЛЯ ПОДАЧИ ЗАЯВОК НА УЧАСТИЕ В СОРЕВНОВАНИЯХ

Пояснительная записка к курсовому проекту

МДК 11.01 Технология разработки и защиты баз данных

|  |  |
| --- | --- |
|  | Руководитель проекта  Р.Ф. Каримова  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 г. |
|  | Студент гр. 21П-1  С.П. Рассказов  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 г. |

2024

Министерство образования и науки Республики Башкортостан

Государственное автономное профессиональное образовательное учреждение

Уфимский колледж статистики, информатики и вычислительной техники

|  |  |
| --- | --- |
|  | УТВЕРЖДАЮ  Заместитель директора  по учебной работе  З.З. Курмашева  « » 2024 г. |

ЗАДАНИЕ

на курсовой проект студенту дневного отделения, группы 21П-1, специальности 09.02.07 Информационные системы и программирование

Фамилия, имя, отчество: Рассказов Сергей Павлович

Тема курсового проекта: «Проектирование базы данных и разработка приложения для подачи заявок на участие в соревнованиях».

Текст задания:

при выполнении курсового проекта должны быть решены следующие задачи:

1. спроектирована структура базы данных;
2. разработана структура программы;
3. реализованы функции авторизации, регистрации, создание новых соревнований, редактирование соревнований, удаление соревнований, поиск соревнований, создание заявок на соревнование, изменение заявок на соревнование, удаление заявок на соревнование, просмотр таблицы игр, просмотр таблицы соревнований, добавление новых спортивных команд, изменение спортивных команд, изменение профиля.

В результате выполнения курсового проекта должны быть представлены:

1. пояснительная записка, состоящая из следующих разделов:

Введение

1 Постановка задачи

2 Экспериментальный раздел

Заключение

Приложения

Список сокращений

Список использованных источников

1. электронный носитель, содержащий разработанный программный продукт;
2. презентация курсового проекта в электронном виде.

Список рекомендуемых источников:

1. Култыгин, О. П. Култыгин, О. П. Администрирование баз данных. СУБД MS SQL Server [Текст] : учеб. пособ. / О. П. Култыгин. - М.: МФПА, 2012. - 232 с.
2. Фуфаев, Э.В. Базы данных [Текст]: учеб. пособ. для студ. учрежд. сред. проф. образования / Э.В. Фуфаев, Д.Э. Фуфаев. - 6-е изд., стер. - М.: Издательский центр «Академия», 2012.- 320 с.- (Среднее профессиональное образование)
3. Википедия [Электронный ресурс] // Свободная энциклопедия. – Режим доступа: <http://ru.wikipedia.org/wiki/>, свободный

Задание к выполнению получил «31» января 2024 г.

Студент Рассказов Сергей Павлович

Срок окончания «31» июня 2024 г.

Руководитель курсового проекта Р.Ф. Каримова

Задание рассмотрено на заседании цикловой комиссии информатики

«11» января 2024 г.

Председатель цикловой комиссии информатики О.В.Фатхулова

Министерство образования и науки Республики Башкортостан

Государственное бюджетное профессиональное образовательное учреждение

Уфимский колледж статистики, информатики и вычислительной техники

ЗАКЛЮЧЕНИЕ

на курсовой проект

Студент Рассказов Сергей Павлович

Группа 21П-1

Специальность 09.02.07 Информационные системы и программирование

Тема Проектирование базы данных и разработка приложения для подачи заявок на участие в соревнованиях

Объем курсового проекта:

количество листов пояснительной записки

количество листов графической части

Заключение о степени соответствия заданию на курсовое проектирование

Характеристика качеств, проявленных студентом при работе над проектом: самостоятельность, дисциплинированность, умение планировать работу и пользоваться литературным материалом и т.д.

Положительные стороны курсового проекта

Недостатки курсового проекта

Характеристика общетехнической и специальной подготовки студента

Заключение и предлагаемая оценка за курсовой проект

Руководитель курсового проекта Каримова Резида Флюновна

«\_\_\_\_» 2024 г.

Подпись

АННОТАЦИЯ

Пояснительная записка к курсовому проекту содержит постановку и программу решения задачи «Проектирование базы данных и разработка приложения для подачи заявок на участие в соревнованиях».

Программа Kursovaya.exe написана на языке C# в среде программирования Visual Studio 2019 с использованием системы управления базой данных MS SQL SERVER предназначена для работы в операционной системе MS Windows 10 и выше, отлажена на данных контрольного примера.

СОДЕРЖАНИЕ

|  |  |
| --- | --- |
|  | лист |
| Введение | 4 |
| 1 Постановка задачи | 4 |
| 1.1 Описание предметной области | 5 |
| 1.2 Описание входной информации | 7 |
| 1.3 Описание выходной информации | 7 |
| 1.4 Концептуальное моделирование | 8 |
| 1.5 Логическое моделирование | 9 |
| 1.6 Описание структуры базы данных | 11 |
| 1.7 Контрольный пример | 14 |
| 1.8 Общие требования к программному продукту | 14 |
| 2 Экспериментальный раздел | 21 |
| 2.1 Описание программы | 21 |
| 2.2 Руководство пользователя | 25 |
| 2.3 Протокол тестирования программного продукта | 34 |
| Заключение | 46 |
| Приложение А «Шаблон выходного документа»  Приложение Б «Схемы баз данных»  Приложение В «Входные данные контрольного примера»  Приложение Г «Выходные данные контрольного примера»  Приложение Д «Код программы» | 33  34  35  38  39 |
| Список источников | 82 |

ВВЕДЕНИЕ

Спортивные соревнования – это увлекательное и захватывающее зрелище, объединяющее людей разных возрастов и национальностей в стремлении к победе и достижению лучших результатов. Они позволяют спортсменам проявить свои навыки, выносливость и дух соперничества, а зрителям – насладиться эмоциями и динамикой соревнований. В мире существует множество различных видов спортивных соревнований – от легких и быстрых беговых дисциплин до сложных и захватывающих командных игр. Для многих спортсменов участие в соревнованиях становится не только способом проверить свои силы, но и возможностью преодолеть себя, улучшить свои результаты и достичь новых высот в спорте.

Актуальность данной работы обусловлена тем, что на данный момент спорт очень популярен среди всех возрастов. Каждый спортсмен мечтает побывать на крупном соревновании в его любимом виде спорта. Спортивные соревнования – достаточно закрытая область, следовательно, у организаторов турниров могут возникать трудности с программным обеспечением для проведения турнира, так как на русском языке я не знаю ни одной программы и приложения для проведения турниров.

Цель курсового проекта – разработка приложения для подачи заявок на соревнования.

Задачами курсового проекта являются:

* описать предметную область;
* разработать структуру базы данных;
* разработать приложение;
* провести тестирование приложения.

1. Постановка задачи

* 1. Описание предметной области

База данных создается для работы приложения, связанной с составлением расписаний игр футбольных команд на соревнования. БД должна содержать данные о соревнованиях, командах, заявках на соревнования, расписание матчей, пользователях.

Приложение предполагает три вида пользователя: клиент, тренер и администратор.

Клиент имеет такие возможности:

* регистрироваться в приложении;
* входить по логину и паролю в приложении;
* просматривать таблицу соревнований;
* просматривать расписание матчей;
* просматривать личный кабинет;
* изменить данные в личном кабинете.

Администратор имеет такие возможности:

* регистрироваться в приложении;
* войти по логину и паролю в приложении;
* создавать соревнование;
* изменять соревнование;
* добавлять соревнование;
* удалять соревнование;
* просматривать личный кабент;
* изменять данные в личном кабинете;
* принимать заявки команд на соревнование;
* изменять заявки команд на соревнование
* удалять заявки команд на соревнование
* просматривать расписание игр;
* создавать команды.

В базе данных должны храниться такие справочники: соревнования, пользователи, команды, заявки.

После создания соревнования, желающие тренеры подают заявку на это соревнование. После чего нужно перейти в специально окно с заявками. Администратор обрабатывает эту заявку и меняет ей статус. Убедиться в том, что команда тренера попала на турнир, можно в отдельном окне с расписанием игр. Если название команды отображается в играх и статус заявки “Обработана”, значит команда попала на соревнование.

В базе данных хранятся такие таблицы:

В таблице “соревнования” содержится следующая информация:

* ID\_соревнования
* Название соревнования;
* Дата начала соревнования;
* Дата окончания соревнования;
* Место проведения соревнования;
* Описание соревнования.

В соревнованиях могут принимать несколько команд (максимум 4). В таблице “команда” содержится следующая информация:

* ID\_команды;
* Название команды;
* Полное ФИО тренера команды.

Команды в программе не изменяются. В таблице “заявка” содержится следующая информация:

* ID\_заявки;
* ID\_соревнования;
* ID\_команды;
* Статус заявки.

Таблица заявка является связующей между таблицами соревнования и команды. В таблице “расписание” содержится следующая информация:

* ID\_расписания;
* Дата расписания;
* Место проведения игр;
* Время начала игр;
* ID\_команды.

Расписание с играми составляется в порядке принятия заявок на игры. В таблице “пользователь” содержится следующая информация:

* ID\_пользователя;
* Полное ФИО пользователя;
* Логин пользователя;
* Пароль пользователя;
* Роль пользователя.

Входными данными являются данные о командах и игроках, турнирах, заявка на участие. Выходными данными являются расписания игр.

1.2 Описание входной информации

Входной информацией для выполнения задачи являются справочники: логины и пароли пользователей, заявка на участие.

1.3 Описание выходной информации

Выходной информацией будет являться документ – расписание соревнования.

Описание выходных документов представлено в таблице 1.3.1.

Таблица 1.3.1 – Описание выходных документов

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Наименование документа (шифр) | Периодичность выдачи документа | Кол-во экз. | Куда передаются | Поля сортировки | Поля группировки | Итоги |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Расписание соревнования | По началу соревнования | 1 | Тренеру | - | - | - |

Шаблон выходного документа представлен в приложении А.

1.4. Концептуальное моделирование

Концептуальная модель базы данных — это некая наглядная диаграмма, нарисованная в принятых обозначениях и подробно показывающая связь между объектами и их характеристиками. Создается концептуальная модель для дальнейшего проектирования базы данных и перевод ее, например, в реляционную базу данных. На концептуальной модели в визуально удобном виде прописываются связи между объектами данных и их характеристиками.

В концептуальной модели есть принятые обозначения элементов. Сущность или объект обозначать прямоугольником, отношения обозначать ромбом, атрибуты объектов, обозначаются овалом.

Концептуальная модель базы данных представлена в приложении Б.

1.5. Логическое моделирование

При логическом моделировании происходит окончательное определение структуры данных, определяются ограничения, накладываемые на эти данные, целью которых является обеспечить целостность данных. Наиболее распространенной моделью данных является реляционная модель. В этой модели данных каждая сущность представляется в виде таблицы.

Логическое моделирование заключается в переходе от концептуальной модели к взаимосвязанным таблицам. Этот переход состоит из следующих шагов:

1. Преобразование сущностей:

- каждая простая сущность становится таблицей.

- каждый атрибут становится столбцом таблицы.

- уникальный идентификатор сущности становится ключом таблицы.

2. Преобразование связи:

- сущности, связанные обязательной связью один к одному можно объединить в одну таблицу.

- связи типа один к одному возможные и связи типа один ко многим реализуются путем переноса ключевых атрибутов таблиц, соответствующих сущностей, стоящих со стороны один в таблице соответствующих сущностей, стоящих со стороны многие.

- связи типа многие ко многим реализуются при помощи промежуточных таблиц, содержащих ключевые атрибуты связываемых таблиц в качестве внешних ключей.

Схема данных – это структура базы данных, описанная на формальном языке, поддерживаемом СУБД (системой управления базы данных). В реляционных базах данных схема определяет таблицы, поля в каждой таблице и ограничения целостности, такие как первичный и внешний ключи.

Схема базы данных представлена в приложении Б.

1.6. Описание структуры базы данных

Описание структуры базы данных представлено в таблице 1.6.1.

Таблица 1.6.1 - Описание структуры базы данных

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поля | Описание поля | Тип данных | Размер поля | Тип ключа[[1]](#footnote-1) |
| 1 | 2 | 3 | 4 | 5 |
| competition (Соревнования) | | | | |
| competition\_Id | ID соревнования | INT |  | PK |
| competition\_Title | Название соревнования | VARCHAR | 50 |  |
| competition\_Date\_Start | Дата начала соревнования | DATETIME |  |  |
| competition\_Date\_Ended | Дата окончания соревнования | DATETIME |  |  |
| competition\_Location | Место проведения соревнования | VARCHAR | 50 |  |
| competition\_Description | Описание соревнования | VARCHAR | 100 |  |
| team (Команда) | | | | |
| team\_Id | ID\_команды | INT |  | PK |
| team\_Title | Название команды | VARCHAR | 50 |  |
| team\_Coach\_FullName | ФИО тренера | VARCHAR | 50 |  |
| request (Заявка) | | | | |
| request\_Id | ID\_заявки | INT |  | PK |
| request\_Competition\_Id | ID\_соревнования | INT |  |  |
| request\_Team\_Id | ID\_команды | INT |  |  |

Продолжение таблицы 1.6.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| timetable (Расписание) | | | | |
| timetable\_Id | ID\_расписания | INT |  | PK |
| timetable\_Date | Дата расписания | DATE |  |  |
| timetable\_Location | Место проведения игр | VARCHAR | 50 |  |
| timetable\_Match\_Started | Время начала игры | DATETIME |  |  |
| timetable\_team\_Id | ID\_команды | INT |  | FK |
| user (Пользователи) | | | | |
| user\_ID | ID\_пользователя | INT |  | PK |
| user\_FullName | ФИО пользователя | VARCHAR | 50 |  |
| user\_Login | Логин пользователя | VARCHAR | 50 |  |
| user\_Password | Пароль пользователя | VARCHAR | 50 |  |
| user\_Role | Роль пользователя | VARCHAR | 6 |  |

1.7. Контрольный пример

Контрольный пример является ручным подсчётом задачи. По составленной программе обрабатываются исходные данные контрольного примера. Полученные результаты сравниваются с известными результатами контрольного примера. При несовпадении результатов производится поиск, исправление ошибок, и снова производится выполнение программы.

Входная информация контрольных примеров представлена в приложении В.

Выходные данные для контрольного примера показан в приложении Г.

1.8 Общие требования к программному продукту

Пользователи должны иметь базовые навыки пользования персональным компьютером.

Минимальные требования к техническому обеспечению программного продукта следующие:

* ОС:  Windows 7 Service Pack1/ Windows 7 64Bit Service Pack1/ Windows 8.1 64Bit / Windows 10 64Bit / Mac OS;
* процессор: 2.40 Ггц (четырехъядерный) / AMD Phenom 9850 (четырехъядерный) @ 2.5 Ггц;
* оперативная память: 512МБ (Win 7/Win8 и выше);
* видеокарта: NVIDIA 9800 GT c 512 МБ видеопамяти/ AMD HD 4870 с 1 Гб видеопамяти (DX 9, 10, 10.1);
* жесткий диск: 10 гигабайт свободного места;
* Microsoft DirectX версия 9.0c;

Функциональные возможности программного продукта:

* приложение должно формировать и отображать выходные данные пользователю;
* в приложении должен быть обеспечен просмотр таблиц (справочников) базы данных с возможность добавления, редактирования, удаления данных.

Требования к надежности:

* приложение должно обрабатывать ошибочные действия пользователя и сообщать ему об этом;
* приложение должно обеспечивать контроль входной и выходной информации.

Требования к информационной и программной совместимости: обеспечить работу приложения с таблицами СУБД MySQL Workbench.

2. Экспериментальный раздел

2.1 Описание программы

Программа имеет модульную структуру. При ее запуске выполняется проект на WPFAuction.exe. Программа Kursovaya.exe написана на языке C# в среде программирования Visual Studio 2022 с использованием системы управления базой данных MS SQL Server 2022.

Описание модулей и методов представлено в таблице 2.1.1.

Рисунок 2.1.1 – Схема взаимодействия модулей

Таблица 2.1.1. - Описание модулей

|  |  |
| --- | --- |
| Методы | Назначение |
| 1 | 2 |
| Auth.xaml.cs | |
| public Auth() | Переход на окно авторизации |
| void authButton\_Click(object sender, RoutedEventArgs e) | Авторизация, если в базе данных есть пользователь с указанными логином и паролем |
| void regButton\_Click(object sender, RoutedEventArgs e) | Переход на окно регистрации |

Продолжение таблицы 2.1.1

|  |  |
| --- | --- |
| 1 | 2 |
| void btnExit\_Click | Завершение работы приложения |
| Registration.xaml.cs | |
| public Registration() | Переход к странице с регистрацией |
| void btnRegistration\_Click(object sender, RoutedEventArgs e) | Регистрация пользователя, если правильно заполнены поля |
| btnBackToAuth\_Click(object sender, RoutedEventArgs e) | Возврат к странице авторизации |
| MatchView.xaml.cs | |
| public MatchView() | Переход к странице с соревнованиями и загрузка данных в таблицу |
| void searchBox\_TextChanged(object sender, TextChangedEventArgs e) | Изменение текста в поле поиска |
| void UpdateCompetitionSearch() | Обновление соревнований в таблице и сравнение их названия с названием соревнования |
| void btnAddCompetition\_Click(object sender, RoutedEventArgs e) | Переход на страницу с добавлением соревнований |
| void btnEditCompetition\_Click(object sender, RoutedEventArgs e) | Переход на страницу с изменением соревнований |
| Image\_MouseDown(object sender, MouseButtonEventArgs e) | Нажатие на иконку профиля и переход в в личный кабинет |

Продолжение таблицы 2.1.1

|  |  |
| --- | --- |
| void btnDeleteCompetition\_Click(object sender, RoutedEventArgs e) | Выбор соревнования в таблице и удаление этого соревнования |
| void searchBox\_TextChanged(object sender, TextChangedEventArgs e) | Изменение текста в поле поиска |
| void UpdateCompetitionSearch() | Обновление соревнований в таблице и сравнение их названия с названием соревнования |
| void btnAddCompetition\_Click(object sender, RoutedEventArgs e) | Переход на страницу с добавлением соревнований |
| void btnEditCompetition\_Click(object sender, RoutedEventArgs e) | Переход на страницу с изменением соревнований |
| Image\_MouseDown(object sender, MouseButtonEventArgs e) | Нажатие на иконку профиля и переход в в личный кабинет |
| void btnDeleteCompetition\_Click(object sender, RoutedEventArgs e) | Выбор соревнования в таблице и удаление этого соревнования |
| void Button\_Click(object sender, RoutedEventArgs e) | Возвращение на окно с авторизацией |
| void btnSendRequest\_Click(object sender, RoutedEventArgs e) | Переход на страницу с отправкой заявки на участие в соревновании |
| void btnAddTeam\_Click(object sender, RoutedEventArgs e) | Добавление команды в базу данных |

Продолжение таблицы 2.1.1

|  |  |
| --- | --- |
| 1 | 2 |
| void searchBox\_TextChanged(object sender, TextChangedEventArgs e) | Изменение текста в поле поиска |
| void UpdateCompetitionSearch() | Обновление соревнований в таблице и сравнение их названия с названием соревнования |
| void btnAddCompetition\_Click(object sender, RoutedEventArgs e) | Переход на страницу с добавлением соревнований |
| void btnEditCompetition\_Click(object sender, RoutedEventArgs e) | Переход на страницу с изменением соревнований |
| Image\_MouseDown(object sender, MouseButtonEventArgs e) | Нажатие на иконку профиля и переход в в личный кабинет |
| void btnDeleteCompetition\_Click(object sender, RoutedEventArgs e) | Выбор соревнования в таблице и удаление этого соревнования |
| void Button\_Click(object sender, RoutedEventArgs e) | Возвращение на окно с авторизацией |
| void btnSendRequest\_Click(object sender, RoutedEventArgs e) | Переход на страницу с отправкой заявки на участие в соревновании |
| void btnAddTeam\_Click(object sender, RoutedEventArgs e) | Добавление команды в базу данных |
| void btnShowTimeTable\_Click(object sender, RoutedEventArgs e) | Поиск номера соревнования и отправка на окно с расписанием матчей |
| AddCompetition.xaml.cs | |
| public AddCompetition() | Переход к окну с добавлением соревнования |
| bool ValidateFields() | Проверка на правильно заполненные данные в полях при создании соревнования |
| void btnAddCompetition\_Click(object sender, RoutedEventArgs e) | Проверка данных в полях и регистрация нового соревнования |
| void tbTitleCompetition\_MouseDoubleClick(object sender, MouseButtonEventArgs e) | Удаление данных из поля “Название соревнования” по двойному клику левой кнопки мыши |
| void btnBackToView\_Click(object sender, RoutedEventArgs e) | Переход на окно с соревнованиями |
| AddTeam.xaml.cs | |
| public AddTeam() | Переход на окно добавления команды |
| bool ValidateFields() | Проверка данных в полях ввода |
| void btnAddTeam\_Click(object sender, RoutedEventArgs e) | Добавление команды в базу данных |
| void btnBackToView\_Click(object sender, RoutedEventArgs e) | Возврат на предыдущее окно |

Продолжение таблицы 2.1.1

|  |  |  |
| --- | --- | --- |
| DeleteCompetition.xaml.cs | | |
| public DeleteCompetition(int findId) | Загрузка страницы удаления соревнования, принятие номера соревнования | |
| void btnYes\_Click(object sender, RoutedEventArgs e) | Нажатие на кнопку “Да” запускает удаление соревнования | |
| void btnNo\_Click(object sender, RoutedEventArgs e) | Нажатие на кнопку “Нет” Возвращает пользователя на предыдущее окно | |
| EditCompetition.xaml.xaml.cs | | |
| public EditCompetition(int findId) | | Загрузка данных об соревновании |
| bool ValidateFields() | | Проверка на корректные данные в полях ввода |
| void btnEditCompetition\_Click(object sender, RoutedEventArgs e) | | Изменение соревнования |
| void btnBackToView\_Click(object sender, RoutedEventArgs e) | | Возвращение на предыдущее окно |
| Profile.xaml.cs | | |
| public Profile() | Загрузка данных о пользователе | |
| void btnEdit\_Click(object sender, RoutedEventArgs e) | Переход на окно с изменением профиля | |
| void btnBackToView\_Click(object sender, RoutedEventArgs e) | Возвращение на предыдущее окно | |

Продолжение таблицы 2.1.1

|  |  |
| --- | --- |
| void btnMyRequest\_Click(object sender, RoutedEventArgs e) | Переход на окно с просмотром заявок |
| EditProfile.xaml.cs | |
| public EditProfile() | Загрузка окна с изменением профиля |
| private bool ValidateFields() | Проверка данных в полях ввода |
| void btnEdit\_Click(object sender, RoutedEventArgs e) | Изменение профиля на значения из полей ввода и перенаправление на предыдущее окно |
| void btnBackToProfile\_Click(object sender, RoutedEventArgs e) | Переход на предыдущее окно |
| Requests.xaml.cs | |
| public Requests() | Загрузка окна с заявками |
| void btnBack\_Click(object sender, RoutedEventArgs e) | Возвращение в предыдущее окно |
| void btnDeleteRequest\_Click(object sender, RoutedEventArgs e) | Переход на окно с подтверждением удаления заявки |
| btnEditRequest\_Click(object sender, RoutedEventArgs e) | Переход на окно с изменением заявки |
| SendRequests.xaml.cs | |
| public SendRequest(int findId) | Загрузка окна с формой отправления заявки и принятие айди соревнования |
| void btnBackToView\_Click(object sender, RoutedEventArgs e) | Возвращение в предыдущее окно |
| void btnSendRequest\_Click(object sender, RoutedEventArgs e) | Отправка измененной заявки в базу данных |
| DeleteRequest.xaml.cs | |
| public DeleteRequest(int findId) | Загрузка окна с удалением заявки |
| void btnYes\_Click(object sender, RoutedEventArgs e) | При нажатии кнопки “Да”, заявка будет удалена и будет осуществлен переход на предыдущее окно |
| void btnNo\_Click(object sender, RoutedEventArgs e) | Возвращение в предыдущее окно |
| TImeTable.xaml.cs | |
| TImeTable(int findId) | Загрузка окна с расписанием матчей |
| void btnBack\_Click(object sender, RoutedEventArgs e) | Возвращение в предыдущее окно |
| Services.cs | |
| public static User GetCurrentUser() | Получение данных о текущем пользователе |
| public static Team GetCurrentTeams() | Получение данных о текущей команде |
| public static List<Team> GetTeams() | Получение данных о командах |
| public static List<User> GetUsers() | Получение данных о пользователях |
| public static List<Competition> GetCompetitions() | Получение данных о соревнованиях |
| public static List<Request> GetRequest() | Получение данных о заявке |
| public static void AddUsers(string Fullname, string Login, string Password, string Role) | Добавление пользователя в базу данных |
| public static void AddTeam(string TitleTeam, string TeamCoachFullname) | Добавление новой команды |
| public static void SendRequest(int RequestId,int numberCompetition, int numberTeam, string Status) | Добавление новой заявки в базу данных |
| public static void AddCompetition(string TitleCompetition, DateTime DateStarted, DateTime DateEnded, string Location, string Description) | Добавление нового соревнования в базу данных |
| public static void UpdateCompetition(int CompetitionId, string newTitleCompetition, DateTime newDateStarted, DateTime newDateEnded, string newLocation, string newDescription) | Изменение заявки в базе данных |
| public static void EditProfile(string newFullName, string newLogin, string newPassword) | Изменение профиля в базе данных |
| public static void EditRequest(int RequestId,int newCompetitionId, int newTeamId, string newStatus) | Измение заявки в базе данных |
| public static void DeleteCompetition(int CompetitionId) | Удаление соревнования из базы данных |
| public static void DeleteRequest(int RequestId) | Удаление заявки из базы данных |

2.2 Протокол тестирования программного продукта

В ходе тестирования программного продукта на корректных и

некорректных данных не было обнаружено ошибок, которые влияли бы на

работу самого программного продукта и всей системы.

Данный программный продукт удовлетворяет всем предъявленным

требованиям, имеет комфортный интерфейс и интуитивно понятный

функционал, исключает появления системных ошибок.

В рисунках 2.2.1 - 2.2.11 представлены протоколы тестирования.

В протоколе тестирования отражаются:

− тестирование на корректных данных;

− тестирование на некорректных данных;

− тестировании продукта на данных контрольного примера.

Тестирование на регистрацию пользователя при корректных данных, ожидаемое сообщение «Вы успешно зарегестрировались.» (рисунок 2.2.1.2).



Рисунок 2.2.1 – Регистрация нового администратора

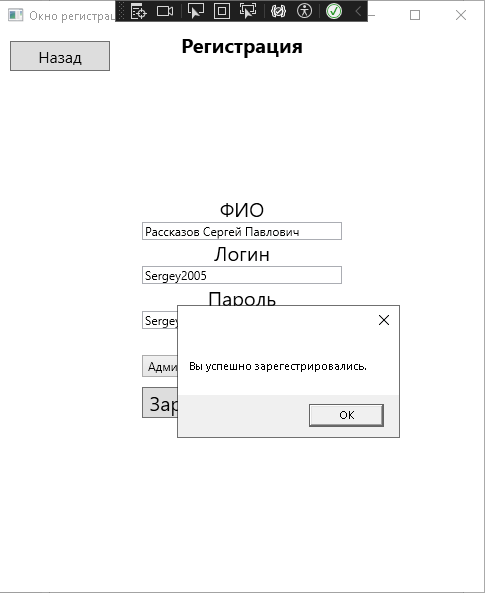


Рисунок 2.2.1.1 – Сообщение об успешной регистрации пользователя

Тестирование авторизации пользователя, ожидается переход на страницу с соревнованиями. (рисунок 2.2.2.2).

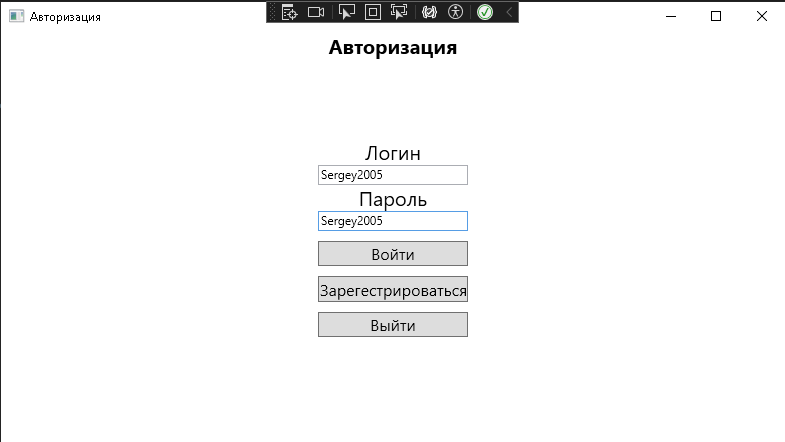


Рисунок 2.2.2 – Ввод данных в поля авторизации

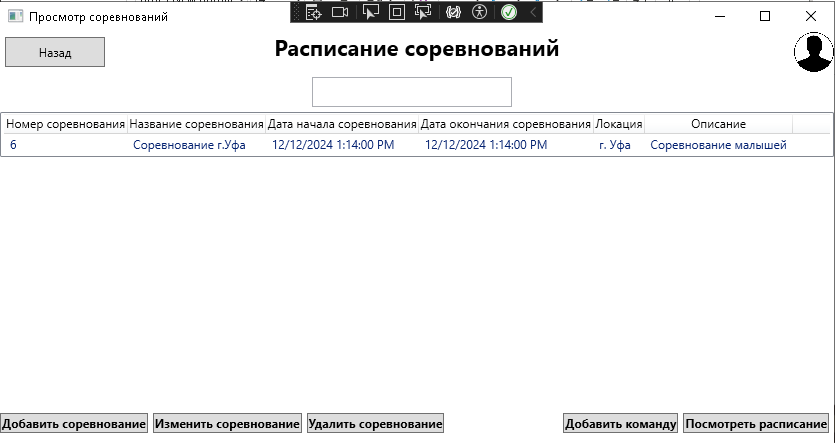


Рисунок 2.2.3 – Страница с соревнованиями

Тестирование добавления нового соревнования представлено на рисунках 2.2.3-2.2.6.

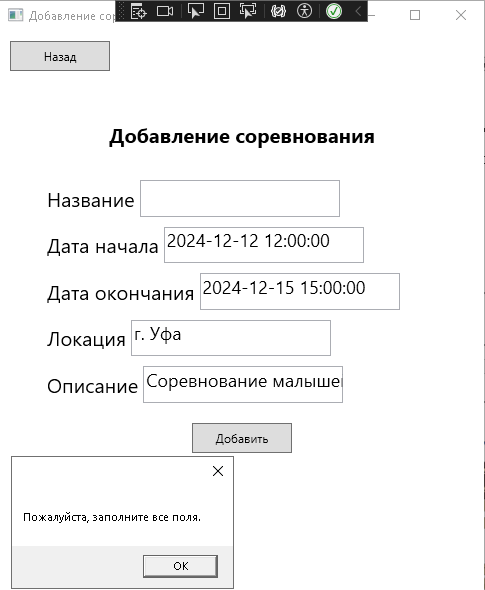


Рисунок 2.2.4 – Просьба заполнить все поля

Тестирование изменение соревнования при вводе корректных данных, ожидаемое сообщение «Соревнование успешно изменено» (рисунок 2.2.7).

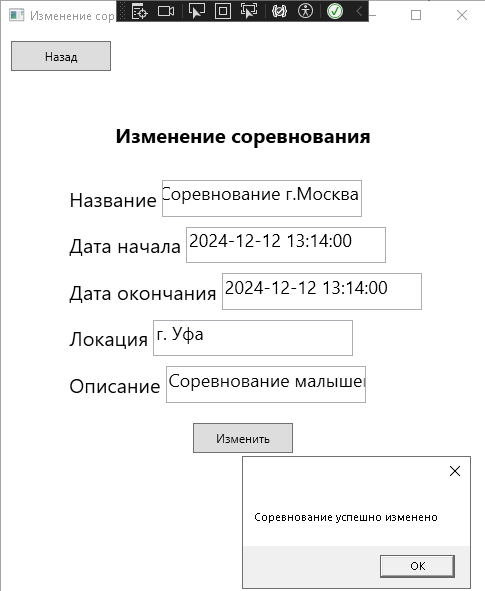


Рисунок 2.2.5 – Сообщение об успешном изменении соревнования

Тестирование изменение профиля при вводе корректных данных, ожидаемое сообщение «Профиль успешно изменен» (рисунок 2.2.8).

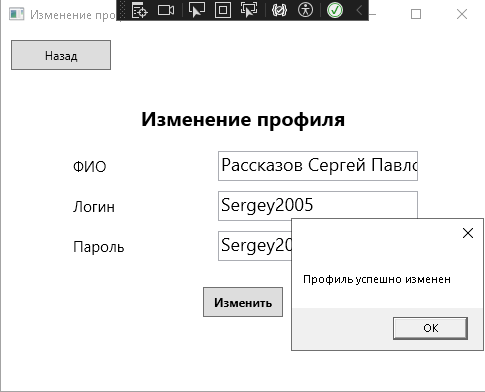


Рисунок 2.2.6 – Сообщение об успешном сохранении данных о ставке

Тестирование удаления соревнования, ожидаемое сообщение «Вы точно хотите удалить соревнование? При нажатии кнопки «Да» происходит удаление и переход на окно с соревнованиями(рисунок 2.2.9-2.2.10).

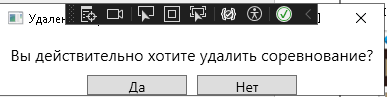


Рисунок 2.2.7 – Сообщение с подтверждением удаления лота

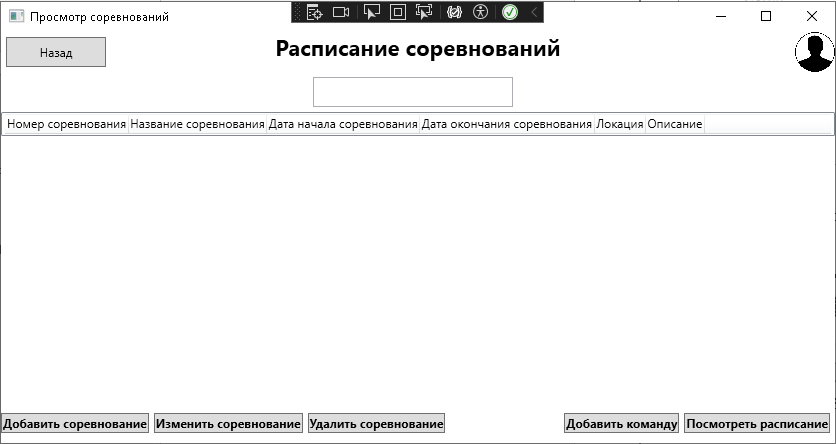


Рисунок 2.2.8 – Соревнование удалилось

Тестирование добавление команды, ожидаемое сообщение «Команда успешно добавлена.» (рисунок 2.2.11).

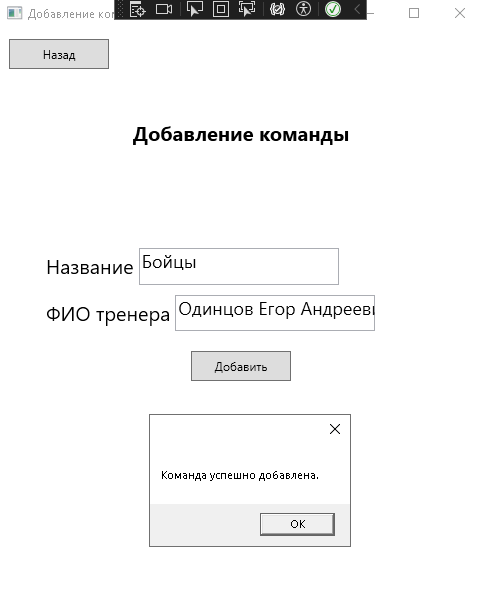


Рисунок 2.2.9 – Сообщение о добавлении новой команды

При тестировании продукта на данных контрольного примера, результаты полностью совпадают с ожидаемыми (рисунки 2.2.12 – 2.2.14)

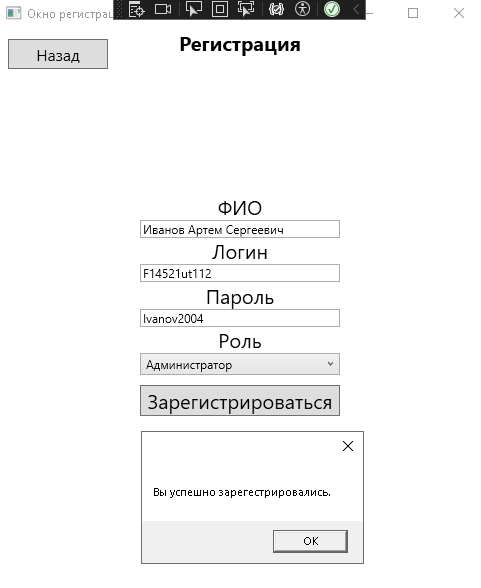
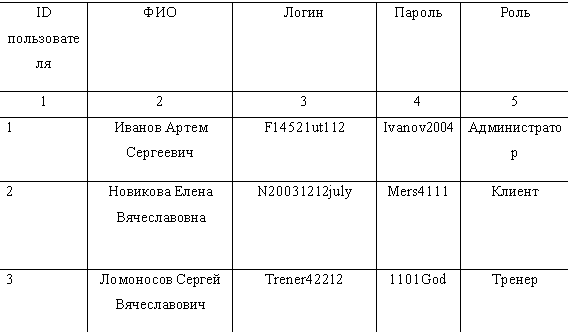


Рисунок 2.2.10 – Регистрация по логину и паролю

Рисунок 2.2.11 – Оформленный список пользователей

2.3 Руководство пользователя

Назначение системы

Программа «Kursovaya» предназначена для для подачи заявок на соревнования, упрощения просмотра клиентом списка соревнований.

Основной целью данной информационной системы является подача заявок на соревнования.

Условия применения системы

Программное обеспечение разрабатывается для персональной вычислительной техники со следующими характеристиками:

* Microsoft Windows 7 / 8 / 10;
* процессор 1 ГГц;
* 128 МБ ОЗУ;
* 60 МБ свободного пространства на диске;
* разрешение экрана монитора не менее 1920 × 1080;
* дополнительное оборудование
* принтер.

Программа «Kursovaya» предназначена для пользователей, имеющих как минимум первоначальные навыки работы с графической операционной системой, на которой будет запускаться данная программа.

Подготовка системы к работе

Для запуска программы необходимо запустить приложение Kursovaya.exe из каталога, в котором установлен данный программный продукт. После этого открывается окно авторизации для входа в программу (рисунок 2.3.1).

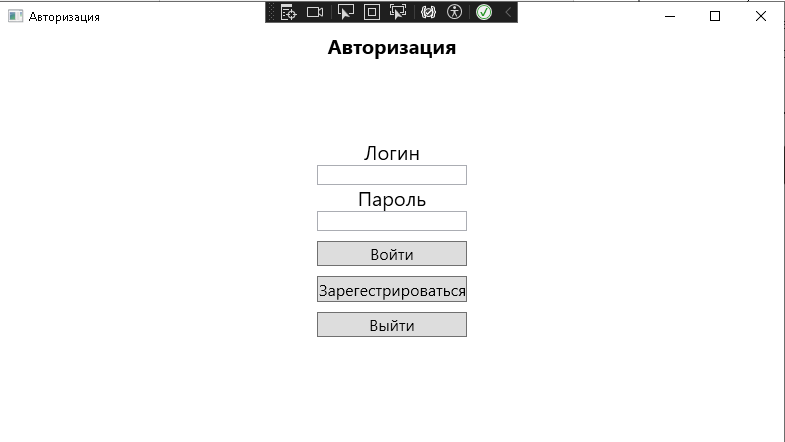


Рисунок 2.3.1 – Окно авторизации для входа в программу

Описание операций

После ввода правильного логина и пароля осуществляется вход в приложение. Если в систему вошел не администратор, то не будут доступны кнопки с функционалом, доступные лишь администратору (рисунки 2.3.2-2.3.3).

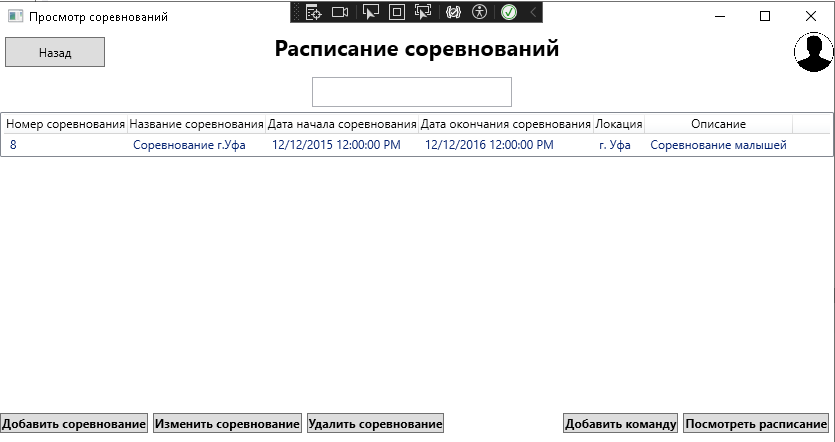


Рисунок 2.3.2 – Интерфейс приложения, в случае входа администратора

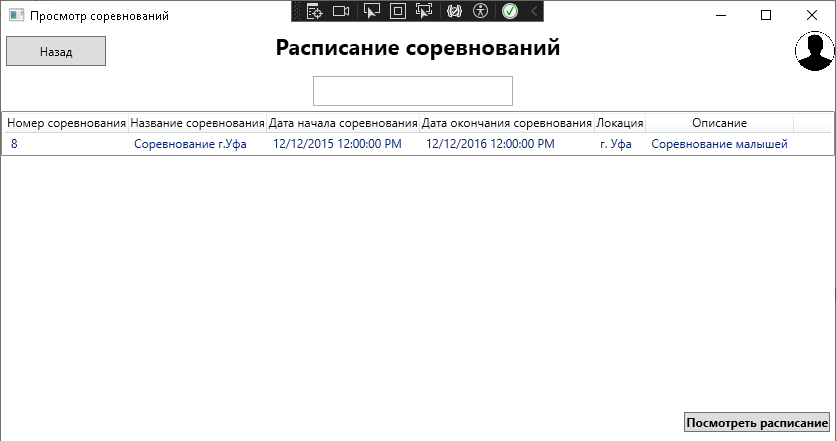


Рисунок 2.3.3 – Интерфейс приложения, в случае входа пользователя

При входе в приложение отображается список соревнований.

Для создания соревнования необходимо зайти с аккаунта администратора и нажать на кнопку «Добавить соревнование», после чего пользователя перенаправит в окно добавления соревнования. (рисунок 2.3.4).

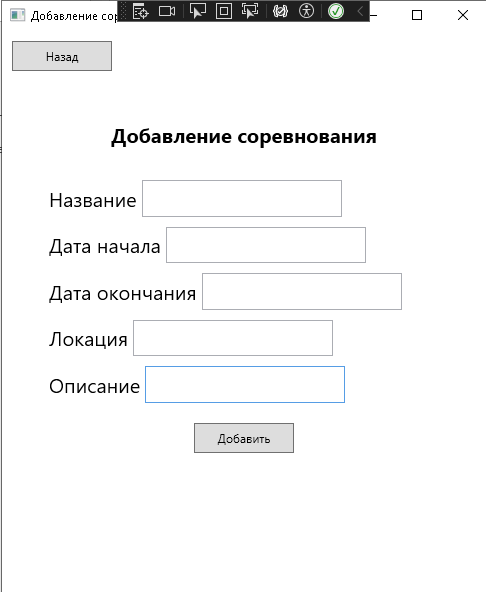


Рисунок 2.3.4 – Интерфейс добавления информации об соревновании

Далее необходимо заполнить все поля, как показано на рисунке 2.3.5, и нажать на кнопку «Добавить».

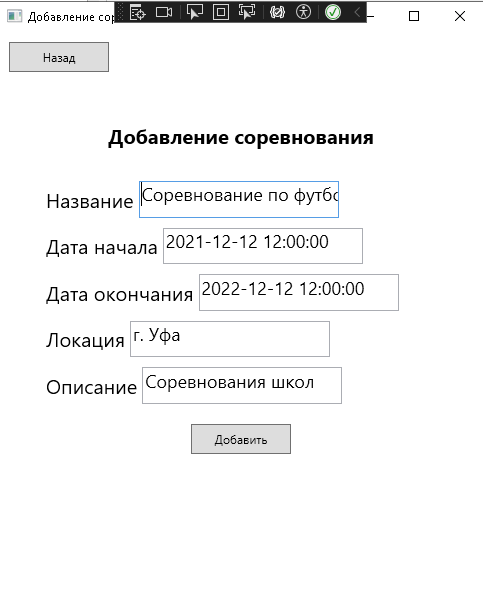


Рисунок 2.3.5 – Добавление информации о соревновании

Для того чтобы вернуться к списку соревнований, необходимо нажать на кнопку «Добавить», после чего соревнование добавится и пользователь вернется к списку с соревнованиями. (рисунок 2.3.6)

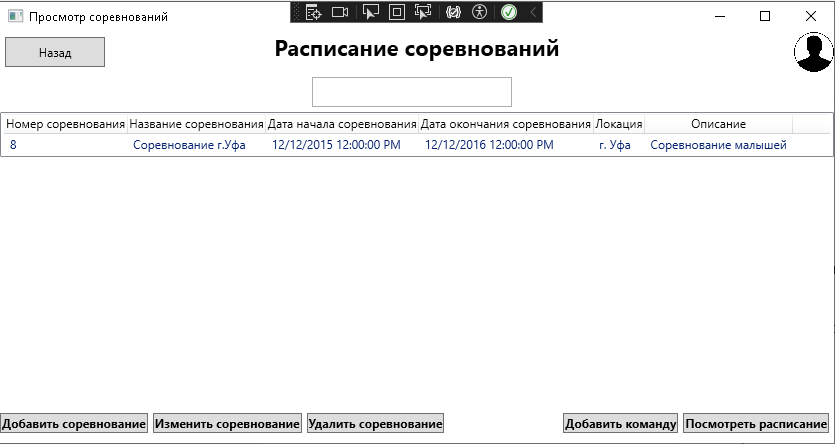


Рисунок 2.3.6 – Список соревнований

Для поиска по каталогу соревнований необходимо ввести запрос в строку поиска. Если введен текст, поиск произойдет по названию соревнования(будет найдено соревнование с введенным именем). (рисунок 2.3.7)

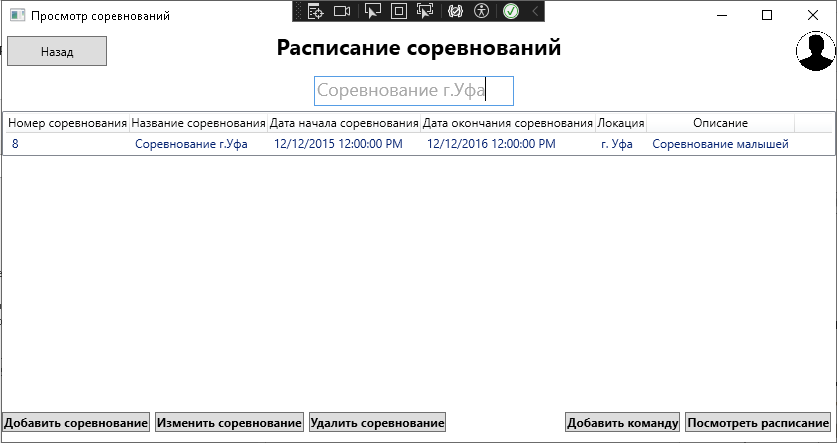


Рисунок 2.3.7 – Панель поиска и поиск соревнования

Для того, чтобы подать заявку, необходимо авторизироваться за роль тренера, нажать на нужное нам соревнование и на кнопку «Подать заявку» (рисунок 2.3.8).

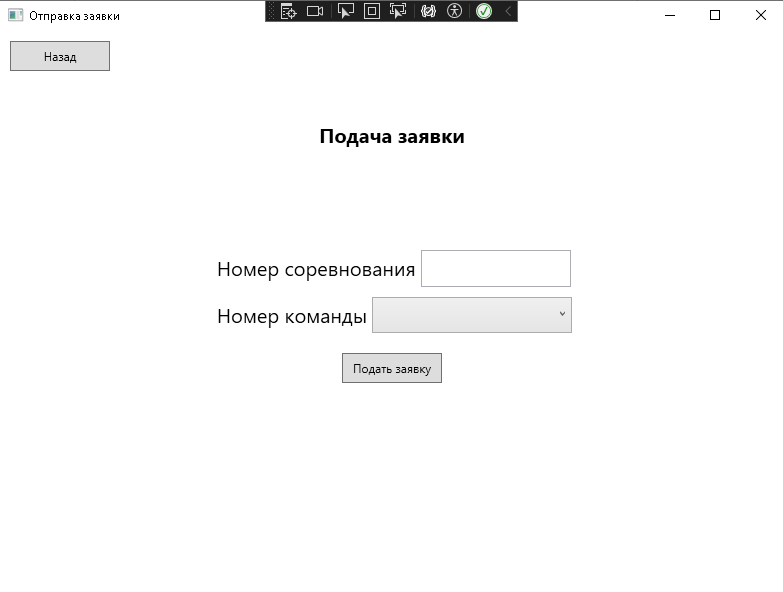


Рисунок 2.3.8 – Подача заявок

Для того, чтобы перейти обратно на страницу с соревнованиями, необходимо нажать на кнопку «Подать заявку» (рисунок 2.3.9).

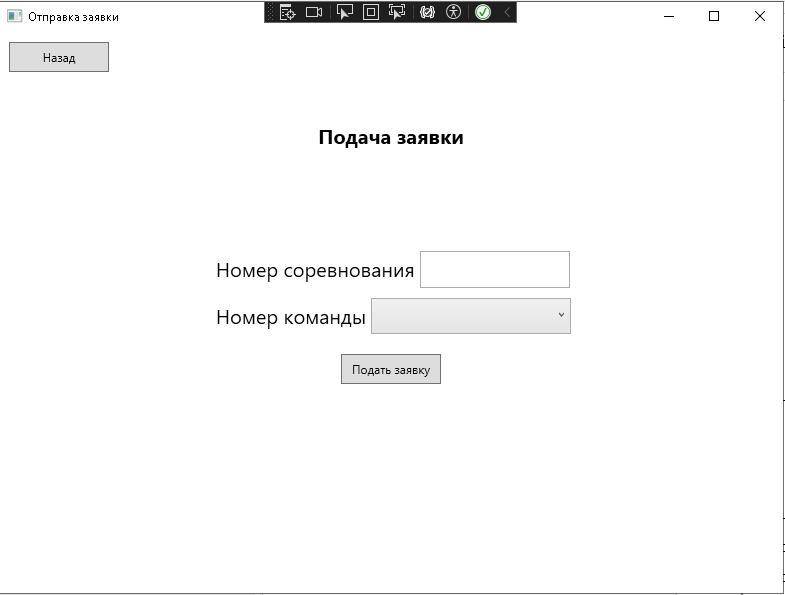


Рисунок 2.3.9 – Подача заявки

Для того, чтобы просмотреть отправленные пользователем заявки, необходимо перейти в личный кабинет и нажать на кнопку «Заявки». Пользователя перенаправит на окно со сделанными заявками. (рисунок 2.3.10).

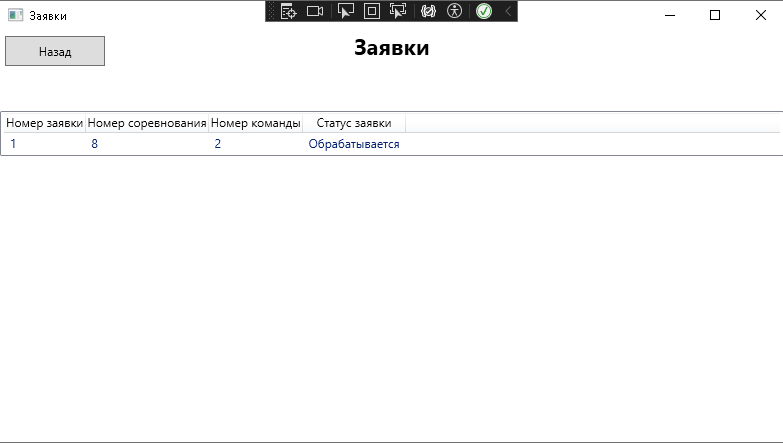


Рисунок 2.3.10 – Окно с заявками

Администратор также может просматривать заявки, сделанные на конкретный лот.

Для того, чтобы изменить личные данные, необходимо нажать на икноку профиля, после чего пользователя перенаправит на окно с личными кабинетом. В личном кабинете нужно нажать на кнопку «Изменить». Для сохранения изменений необходимо нажать на кнопку «Изменить», после чего произодет переход на предыдущее окно. (рисунки 2.3.11).

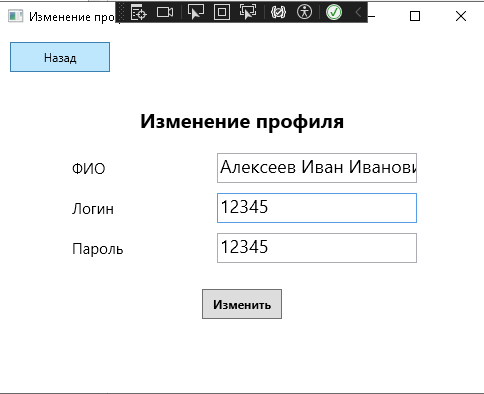


Рисунок 2.3.11 – Страница «Информация о пользователе»

Чтобы выйти из системы пользователь должен выйти на страницу с соревнованиями и нажать на кнопку «Назад», после чего пользователя пренаправит на страницу входа.

ЗАКЛЮЧЕНИЕ

В процессе выполнения курсового проекта были разработаны структура и алгоритм работы WPF-приложения «Подача заявок на соревнования».

При этом были изучены особенности реализации компонентов WPF для построения клиентских приложений с взаимодействием с пользователем.

Результатом работы стало создание WPF-приложения для подачи заявок на соревнования.

WPF-приложение написано на языке C# в среде разработки Visual Studio 2022 с использованием языка разметки XAML и системы управления базой данных MS SQL Server 2022.

Были проведены опытная эксплуатация и отладочное тестирование WPF приложения. По результатам отладочного тестирования были устранены некоторые недостатки, в частности были обнаружены и исправлены неточности в реализации алгоритмов. После этого было написано руководство пользователя.

С помощью приложения на основании данных контрольного примера были получены результаты, которые полностью совпадают с выходной информацией контрольного примера.

Шаблон выходного документа

{Название соревнования}

Дата начала соревнования: {Дата начала соревнования}.

Дата окончания соревнования: {Дата окончания соревнования}.

Адрес: {Место проведения соревнования}.

Описание: {Описание}.

Команда 1: {Название команды}, {ФИО тренера команды}.

Команда 2: {Название команды}, {ФИО тренера команды}.

Команда 3: {Название команды}, {ФИО тренера команды}.

Команда 4: {Название команды}, {ФИО тренера команды}.

Команда 5: {Название команды}, {ФИО тренера команды}.

Команда 6: {Название команды}, {ФИО тренера команды}.

Рисунок А.1 – Расписание соревнования

Концептуальная модель БД

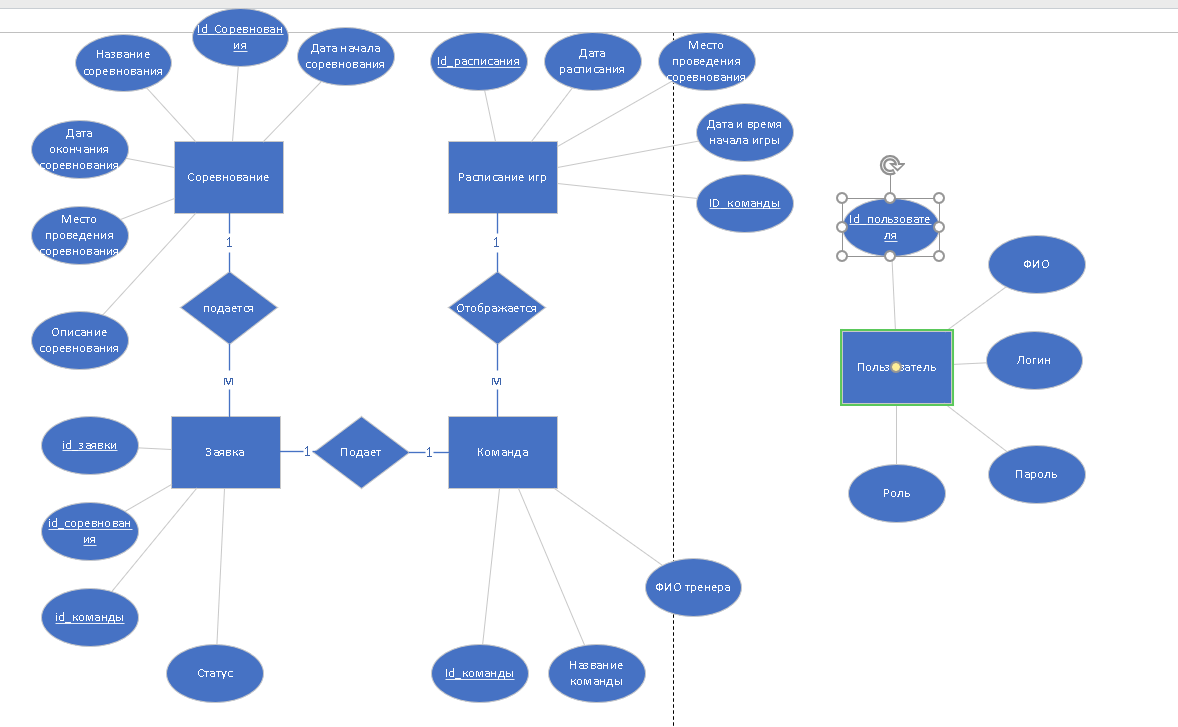


Рисунок Б.1 – Концептуальная модель БД

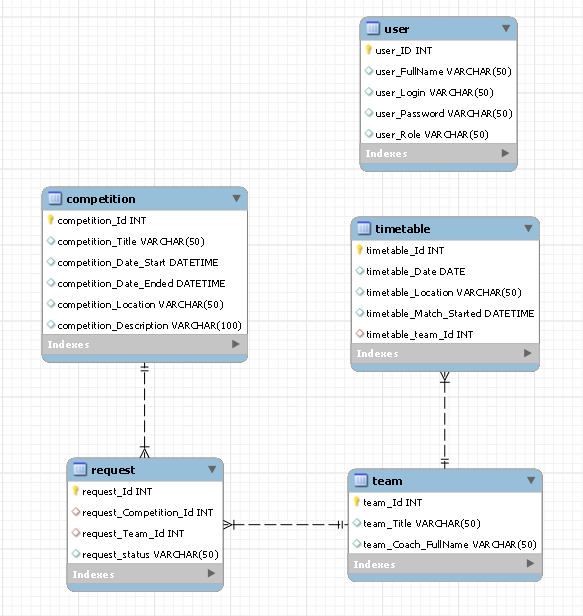


Рисунок Б.2 – Схема данных БД

Входные данные контрольного примера

Таблица В.1 – список команд

|  |  |
| --- | --- |
| Название команды | ФИО тренера |
| Ворота | Криштиану Роналду душ Сантуш Авейру |
| Девятка | Лионель Андрес Месси Куччитини |
| Центр | Игорь Владимирович Акинфеев |
| Уфа | Петр Чех |
| Мяч | Давид Хосе Хименес Сильва |
| Сармат | Дэвид Ро́берт Джо́зеф Бе́кхэм |

Таблица В.2 – Список соревнований

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название соревнования | Дата начала соревнования | Дата окончания соревнования | Место проведения | Описание соревнования |
| Кубок Америки — 2024 | 21.06.2024 | 15.07.2024 | Хард Рок (Майами, Флорида, США) | Кубок Америки по футболу 2024 — 48-й розыгрыш главного мужского футбольного турнира среди южноамериканских и североамериканских команд, организуемого КОНМЕБОЛ, который проходил с 20 июня по 14 июля 2024 года в США. |

Продолжение таблицы В.2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ЧМ-2026 — Азия | 12.10.2023 | 15.11.2025 | Робина (Голд-Кост, Австралия) | Главное международное соревнование по футболу, проводимое с 1930 года. Чемпионат мира организуется управляющим органом мирового футбола — ФИФА, и участвовать в нём могут мужские национальные сборные стран-членов ФИФА всех континентов. |
| Товарищеские матчи U21 — 2024 | 01.01.2024 | 31.12.2024 | Роналдо Луис Назарио де Лима (Рио-де-Жанейро, Бразилия) | Товарищеские матчи U21 часто проводятся в рамках подготовки к чемпионатам Европы, мира и олимпийским играм. Они могут включать как традиционные футбольные нации, так и менее известные команды, что позволяет игрокам показать свои навыки на международной арене. |

Таблица В.4 – список логинов и паролей пользователей

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID пользователя | ФИО | Логин | Пароль | Роль |
| 1 | 2 | 3 | 4 | 5 |
| 1 | Иванов Артем Сергеевич | F14521ut112 | Ivanov2004 | Администратор |
| 2 | Новикова Елена Вячеславовна | N20031212july | Mers4111 | Клиент |
| 3 | Ломоносов Сергей Вячеславович | Trener42212 | 1101God | Тренер |

Выходные данные контрольного примера

ЧМ-2026 — Азия

Дата начала соревнования: 12.10.2023

Дата окончания соревнования: 15.11.2025

Адрес: Робина (Голд-Кост, Австралия)

Описание: Главное международное соревнование по футболу, проводимое с 1930 года. Чемпионат мира организуется управляющим органом мирового футбола — ФИФА, и участвовать в нём могут мужские национальные сборные стран-членов ФИФА всех континентов.

Команда 1: Ворота, Криштиану Роналду душ Сантуш Авейру.

Команда 2: Девятка, Лионель Андрес Месси Куччитини.

Команда 3: Центр, Игорь Владимирович Акинфеев.

Команда 4: Уфа, Петр Чех.

Команда 5: Мяч, Давид Хосе Хименес Сильва.

Команда 6: Сармат, Дэвид Ро́берт Джо́зеф Бе́кхэм.

Рисунок А.1 – Расписание соревнования

Код программы

Форма MainWindow.xaml.cs (Рабочее окно)

using Auction.Classes;

using Auction.Views;

using System.Data;

using System.Windows;

using System.Windows.Navigation;

namespace Auction

{

/// <summary>

/// Логика взаимодействия для MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

MainFrame.Navigate(new SignInPage());

Manager.MainFrame = MainFrame;

btnAuctions.Visibility = Visibility.Hidden;

btnUserBids.Visibility = Visibility.Hidden;

btnUserSettings.Visibility = Visibility.Hidden;

btnSignOut.Visibility = Visibility.Hidden;

}

private void BtnUserSettings\_Click(object sender, RoutedEventArgs e)

{

Manager.MainFrame.Navigate(new UserInfoPage());

}

private void BtnAuctions\_Click(object sender, RoutedEventArgs e)

{

while (Manager.MainFrame.CanGoBack)

Manager.MainFrame.GoBack();

Manager.MainFrame.GoForward();

}

private void BtnUserBids\_Click(object sender, RoutedEventArgs e)

{

Manager.MainFrame.Navigate(new BidsPage());

}

private void MainFrame\_Navigating(object sender, NavigatingCancelEventArgs e)

{

if(SignInPage.userInfo.Rows.Count!=0)

{

tbUserName.Text = SignInPage.userInfo.Rows[0][1] + " " + SignInPage.userInfo.Rows[0][2] +

"\n" + SignInPage.userInfo.Rows[0][3];

btnAuctions.Visibility = Visibility.Visible;

btnUserBids.Visibility = Visibility.Visible;

btnUserSettings.Visibility = Visibility.Visible;

btnSignOut.Visibility = Visibility.Visible;

}

else

{

tbUserName.Text="";

btnAuctions.Visibility = Visibility.Hidden;

btnUserBids.Visibility = Visibility.Hidden;

btnUserSettings.Visibility = Visibility.Hidden;

btnSignOut.Visibility = Visibility.Hidden;

}

}

private void BtnSignOut\_Click(object sender, RoutedEventArgs e)

{

MessageBoxResult confirm = MessageBox.Show("Вы уверены, что хотите выйти?", "Подтверждение", MessageBoxButton.YesNo);

if (confirm == MessageBoxResult.Yes)

{

SignInPage.userInfo = new DataTable();

while (MainFrame.CanGoBack)

MainFrame.GoBack();

}

}

}

}

Форма SignIn.xaml.cs (Вход в систему)

using Auction.Classes;

using System.Data;

using System.Windows;

using System.Windows.Controls;

namespace Auction.Views

{

/// <summary>

/// Логика взаимодействия для SignInPage.xaml

/// </summary>

public partial class SignInPage : Page

{

public static bool isAdmin;//хранит информацию о том, вошел ли в систему админ

public static DataTable userInfo = new DataTable();

public SignInPage()

{

InitializeComponent();

tbLogin.Focus();

}

private void BtnSignIn\_Click(object sender, RoutedEventArgs e)

{

if (checkIsShowPassword.IsChecked == true)

{

if (tbLogin.Text == "" && tbPassword.Text == "")

MessageBox.Show("Введите логин и пароль!");

else if (tbLogin.Text == "")

MessageBox.Show("Введите логин!");

else if (tbPassword.Text == "")

MessageBox.Show("Введите пароль!");

else

{

DB.LoadDataTable(userInfo, "select \* from users where login='" + tbLogin.Text + "' and password='" + tbPassword.Text + "' and isadmin=1");

if (userInfo.Rows.Count > 0)

{

isAdmin = true;

Manager.MainFrame.Navigate(new AuctionsPage());

}

else

{

DB.LoadDataTable(userInfo, "select \* from users where login='" + tbLogin.Text + "' and password='" + tbPassword.Text + "' and isadmin=0");

if (userInfo.Rows.Count > 0)

{

isAdmin = false;

Manager.MainFrame.Navigate(new AuctionsPage());

}

else

{

MessageBox.Show("Введен неправильный логин или пароль!");

}

}

}

}

else

{

if (tbLogin.Text == "" && pbPassword.Password == "")

MessageBox.Show("Введите логин и пароль!");

else if (tbLogin.Text == "")

MessageBox.Show("Введите логин!");

else if (pbPassword.Password == "")

MessageBox.Show("Введите пароль!");

else

{

DB.LoadDataTable(userInfo, "select \* from users where login='" + tbLogin.Text + "' and password='" + pbPassword.Password + "' and isadmin=1");

if (userInfo.Rows.Count > 0)

{

isAdmin = true;

Manager.MainFrame.Navigate(new AuctionsPage());

}

else

{

DB.LoadDataTable(userInfo, "select \* from users where login='" + tbLogin.Text + "' and password='" + pbPassword.Password + "' and isadmin=0");

if (userInfo.Rows.Count > 0)

{

isAdmin = false;

Manager.MainFrame.Navigate(new AuctionsPage());

}

else

{

MessageBox.Show("Введен неправильный логин или пароль!");

}

}

}

}

}

private void CheckIsShowPassword\_Checked(object sender, RoutedEventArgs e)

{

tbPassword.Text = pbPassword.Password;

pbPassword.Visibility = Visibility.Hidden;

tbPassword.Visibility = Visibility.Visible;

}

private void CheckIsShowPassword\_Unchecked(object sender, RoutedEventArgs e)

{

pbPassword.Password = tbPassword.Text;

pbPassword.Visibility = Visibility.Visible;

tbPassword.Visibility = Visibility.Hidden;

}

private void BtnRegistr\_Click(object sender, RoutedEventArgs e)

{

Manager.MainFrame.Navigate(new UserInfoPage());

}

}

}

Форма AuctionsPage.xaml.cs (Аукционы)

using Auction.Classes;

using System;

using System.Collections.Generic;

using System.Data;

using System.Windows;

using System.Windows.Controls;

namespace Auction.Views

{

/// <summary>

/// Логика взаимодействия для AuctionsPage.xaml

/// </summary>

public partial class AuctionsPage : Page

{

List<AuctionControl> futureAuctions = new List<AuctionControl>();

List<AuctionControl> pastAuctions = new List<AuctionControl>();

public AuctionsPage()

{

InitializeComponent();

LoadInfo();

}

public void LoadInfo()

{

futureAuctions.Clear();

pastAuctions.Clear();

if (!SignInPage.isAdmin)

{

btnAddAuction.Visibility = Visibility.Hidden;

}

DataTable dbAuctions = new DataTable();

DB.LoadDataTable(dbAuctions, "select \* from auctions");

foreach (DataRow row in dbAuctions.Rows)

{

if (DateTime.Parse(row[5].ToString()) >= DateTime.Now)

{

futureAuctions.Add(new AuctionControl(row, true));

}

else

{

pastAuctions.Add(new AuctionControl(row, false));

}

}

lvAuctions.ItemsSource = futureAuctions;

}

private void BtnFutureAuctions\_Click(object sender, RoutedEventArgs e)

{

lvAuctions.ItemsSource = futureAuctions;

}

private void BtnPastAuctions\_Click(object sender, RoutedEventArgs e)

{

lvAuctions.ItemsSource = pastAuctions;

}

private void BtnAddAuction\_Click(object sender, RoutedEventArgs e)

{

Manager.MainFrame.Navigate(new AuctionCatalogPage());

}

private void BtnRemoveAuction\_Click(object sender, RoutedEventArgs e)

{

MessageBoxResult confirm = MessageBox.Show(

"Вы точно хотите удалить аукцион? После удаления данные нельзя будет восстановить",

"Подтверждение", MessageBoxButton.YesNo, MessageBoxImage.Exclamation);

if(confirm==MessageBoxResult.Yes)

{

int id = Convert.ToInt32((lvAuctions.SelectedItem as DataRowView)[0].ToString());

if (DateTime.Parse((lvAuctions.SelectedItem as DataRowView)[5].ToString()) >= DateTime.Now)

futureAuctions.Remove(futureAuctions.Find(item => (int)item.Tag == id));

else

pastAuctions.Remove(futureAuctions.Find(item => (int)item.Tag == id));

string sql = "delete from auctions where auction\_no="+id;

DB.QueryData(sql);

MessageBox.Show("Удалено успешно!");

}

}

private void Grid\_Loaded(object sender, RoutedEventArgs e)

{

LoadInfo();

}

}

}

Форма AuctionCatalogPage.xaml (Каталог аукциона)

using Auction.Classes;

using Microsoft.Win32;

using System;

using System.Collections.Generic;

using System.Data;

using System.IO;

using System.Reflection;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Media;

using Word = Microsoft.Office.Interop.Word;

namespace Auction.Views

{

/// <summary>

/// Логика взаимодействия для AuctionCatalogPage.xaml

/// </summary>

public partial class AuctionCatalogPage : System.Windows.Controls.Page

{

List<LotControl> lotControls = new List<LotControl>();

DataTable dataAuctionCatalog = new DataTable();

byte[] imageData = null;

public static string aucNo = "";

public static bool isFuture;

int countSelected;

int selectedPage = 1;

int countPages;

public static int countRows;

bool isChanged;

public AuctionCatalogPage(DataRow row, bool b)

{

InitializeComponent();

if (!SignInPage.isAdmin)

{

btnAddImage.Visibility = Visibility.Hidden;

btnAddLot.Visibility = Visibility.Hidden;

btnSaveChanges.Visibility = Visibility.Hidden;

btnDeleteAuction.Visibility = Visibility.Hidden;

btnGetBids.Visibility = Visibility.Collapsed;

dpAuctionDate.Visibility = Visibility.Hidden;

spDateTime.Visibility = Visibility.Hidden;

tbAuctionName.IsReadOnly = true;

tbAuctionName.IsReadOnlyCaretVisible = false;

tbAuctionName.Background = new SolidColorBrush(Colors.Transparent);

tbAuctionName.BorderThickness = new Thickness(0);

tbAuctionDate.IsReadOnly = true;

tbAuctionDate.IsReadOnlyCaretVisible = false;

tbAuctionDate.Background = new SolidColorBrush(Colors.Transparent);

tbAuctionDate.BorderThickness = new Thickness(0);

tbAuctionPlace.IsReadOnly = true;

tbAuctionPlace.IsReadOnlyCaretVisible = false;

tbAuctionPlace.Background = new SolidColorBrush(Colors.Transparent);

tbAuctionPlace.BorderThickness = new Thickness(0);

tbAuctionDescription.IsReadOnly = true;

tbAuctionDescription.IsReadOnlyCaretVisible = false;

tbAuctionDescription.Background = new SolidColorBrush(Colors.Transparent);

tbAuctionDescription.BorderThickness = new Thickness(0);

}

else

{

tbAuctionDate.Width = 0;

dpAuctionDate.SelectedDate = DateTime.Parse(row[5].ToString());

tbHour.Text = DateTime.Parse(row[5].ToString()).Hour.ToString();

tbMinute.Text = DateTime.Parse(row[5].ToString()).Minute.ToString();

}

aucNo = row[0].ToString();

tbAuctionName.Text = row[1].ToString();

tbAuctionPlace.Text = row[2].ToString();

tbAuctionDescription.Text = row[3].ToString();

tbAuctionDate.Text = row[5].ToString();

btnOpenCloseDescribe.Click += new RoutedEventHandler(BtnOpenDescription\_Click);

dpAuctionDate.SelectedDate = DateTime.Now;

countRows = (int)DB.QueryScalar("select count(\*) from lots where auction\_no=" + aucNo);

isFuture = b;

cbSort.SelectedIndex = 0;

isChanged = false;

}

public AuctionCatalogPage()

{

InitializeComponent();

tbAuctionDate.Width = 0;

tbAuctionName.Height = 100;

btnOpenCloseDescribe.Click += new RoutedEventHandler(BtnOpenDescription\_Click);

tbAuctionName.Width = 700;

tbAuctionDescription.Width = 700;

tbAuctionDescription.Height = 100;

}

private void BtnBackToAuctions\_Click(object sender, RoutedEventArgs e)

{

if (isChanged)

{

MessageBoxResult confirm = MessageBox.Show(

"Вы уверены что хотите выйти? Все несохраненные данные будут утеряны",

"Подтверждение", MessageBoxButton.YesNo, MessageBoxImage.Question);

if (confirm == MessageBoxResult.Yes)

{

Manager.MainFrame.GoBack();

aucNo = "";

}

}

else

{

Manager.MainFrame.GoBack();

aucNo = "";

}

}

private void BtnOpenDescription\_Click(object sender, RoutedEventArgs e)

{

tbAuctionName.Visibility = Visibility.Collapsed;

tbAuctionDate.Visibility = Visibility.Collapsed;

tbAuctionPlace.Visibility = Visibility.Collapsed;

dpAuctionDate.Visibility = Visibility.Collapsed;

spDateTime.Visibility = Visibility.Collapsed;

tbAuctionDescription.Visibility = Visibility.Visible;

btnOpenCloseDescribe.Content = "Закрыть описание";

btnOpenCloseDescribe.Click -= new RoutedEventHandler(BtnOpenDescription\_Click);

btnOpenCloseDescribe.Click += new RoutedEventHandler(BtnCloseDescription\_Click);

}

private void BtnCloseDescription\_Click(object sender, RoutedEventArgs e)

{

tbAuctionName.Visibility = Visibility.Visible;

tbAuctionDate.Visibility = Visibility.Visible;

tbAuctionPlace.Visibility = Visibility.Visible;

if (!SignInPage.isAdmin)

{

dpAuctionDate.Visibility = Visibility.Collapsed;

spDateTime.Visibility = Visibility.Collapsed;

}

else

{

dpAuctionDate.Visibility = Visibility.Visible;

spDateTime.Visibility = Visibility.Visible;

}

tbAuctionDescription.Visibility = Visibility.Collapsed;

btnOpenCloseDescribe.Content = "Описание аукциона";

btnOpenCloseDescribe.Click -= new RoutedEventHandler(BtnCloseDescription\_Click);

btnOpenCloseDescribe.Click += new RoutedEventHandler(BtnOpenDescription\_Click);

}

private void BtnAddLot\_Click(object sender, RoutedEventArgs e)

{

if (aucNo == "")

MessageBox.Show("Сначала создайте аукцион");

else

Manager.MainFrame.Navigate(new LotPage());

}

private void BtnDeleteLot\_Click(object sender, RoutedEventArgs e)

{

MessageBoxResult confirm = MessageBox.Show(

"Вы точно хотите удалить лот? После удаления данные нельзя будет восстановить.",

"Подтверждение", MessageBoxButton.YesNo, MessageBoxImage.Question);

if (confirm == MessageBoxResult.Yes)

{

string id = (lvLotList.SelectedItem as DataRowView)[0].ToString();

string lotNo = (lvLotList.SelectedItem as DataRowView)[3].ToString();

string sql = "delete from lot where id\_lot=" + id;

DB.QueryData(sql);

sql = string.Format("update lot set lot\_no=(select l.lot\_no-1 from lot as l where auction\_no={0}) " +

"where auction\_no={0} and lot\_no>{1}", aucNo, lotNo);

DB.QueryData(sql);

MessageBox.Show("Удалено успешно!");

LoadLotsWithFilters();

}

}

private void BtnSaveChanges\_Click(object sender, RoutedEventArgs e)

{

if (tbAuctionName.Text == "Название аукциона")

MessageBox.Show("Введите название аукциона");

else if (dpAuctionDate.Text == "Выбор даты")

MessageBox.Show("Выберите дату проведения");

else if (tbHour.Text == "" || tbMinute.Text == "")

MessageBox.Show("Введите время проведения аукциона");

else if (tbAuctionPlace.Text == "Место проведения")

MessageBox.Show("Введите адрес проведения аукциона");

else if (tbAuctionDescription.Text == "")

MessageBox.Show("Введите описание аукциона. Для того, " +

"чтобы открыть панель описания, нажмите кнопку \"Описание аукциона\".");

else

{

DateTime date = dpAuctionDate.SelectedDate.Value;

TimeSpan ts = new TimeSpan(Convert.ToInt32(tbHour.Text), Convert.ToInt32(tbMinute.Text), 0);

date = date.Date + ts;

string sqlFormattedDate = date.ToString("yyyy-dd-MM HH:mm:ss.fff");

string sql;

if (aucNo == "")

{

sql = string.Format("insert into auctions(" +

"auction\_no,auction\_name,auction\_place,auction\_description,auction\_date) values((" +

"select max(a.auction\_no) from auctions as a)+1,'{0}','{1}','{2}','{3}')",

tbAuctionName.Text, tbAuctionPlace.Text, tbAuctionDescription.Text, sqlFormattedDate);

}

else

{

sql = string.Format("update auctions set auction\_name='{0}',auction\_place='{1}'," +

"auction\_description='{2}',auction\_date='{3}' where auction\_no={4}",

tbAuctionName.Text, tbAuctionPlace.Text, tbAuctionDescription.Text, sqlFormattedDate, aucNo);

}

DB.QueryData(sql);

if (aucNo == "")

{

sql = string.Format("select auction\_no from auctions where auction\_name like '{0}' and auction\_place like '{1}' and " +

"auction\_description like '{2}' and auction\_date='{3}'",

tbAuctionName.Text, tbAuctionPlace.Text, tbAuctionDescription.Text, sqlFormattedDate);

aucNo = DB.QueryScalar(sql).ToString();

}

if (imageData != null)

{

sql = "update auctions set auction\_image=@image where auction\_no=" + aucNo;

DB.QueryImage(imageData, sql);

}

MessageBox.Show("Выполнено успешно!");

}

}

private void BtnDeleteAuction\_Click(object sender, RoutedEventArgs e)

{

MessageBoxResult confirm = MessageBox.Show(

"Вы точно хотите удалить аукцион? После удаления данные нельзя будет восстановить.",

"Подтверждение", MessageBoxButton.YesNo, MessageBoxImage.Question);

if (confirm == MessageBoxResult.Yes)

{

string sql = "delete from auctions where auction\_no=" + aucNo;

DB.QueryData(sql);

MessageBox.Show("Успешно!");

Manager.MainFrame.GoBack();

}

}

private void BtnAddImage\_Click(object sender, RoutedEventArgs e)

{

OpenFileDialog fd = new OpenFileDialog();

if (fd.ShowDialog() == true)

{

using (FileStream fs = new FileStream(fd.FileName, FileMode.Open))

{

imageData = new byte[fs.Length];

fs.Read(imageData, 0, imageData.Length);

}

MessageBox.Show("Изображение добавлено. Чтобы " +

"сохранить изменения, нажмите на кнопку \"Сохранить\"");

}

}

private void CbSort\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

selectedPage = 1;

LoadLotsWithFilters();

}

private void BtnPageDown\_Click(object sender, RoutedEventArgs e)

{

if (selectedPage > 1)

{

selectedPage--;

LoadLotsWithFilters();

}

}

private void BtnPageUp\_Click(object sender, RoutedEventArgs e)

{

if (selectedPage < countPages)

{

selectedPage++;

LoadLotsWithFilters();

}

}

private void BtnPage\_Click(object sender, RoutedEventArgs e)

{

selectedPage = Convert.ToInt32(((Button)sender).Content);

LoadLotsWithFilters();

}

private void TbHour\_PreviewTextInput(object sender, System.Windows.Input.TextCompositionEventArgs e)

{

if (!char.IsDigit(e.Text, 0) || tbHour.Text.Length > 2 ||

(tbHour.Text.Length > 0 && Convert.ToInt32(tbHour.Text) >= 24))

e.Handled = true;

}

private void TbMinute\_PreviewTextInput(object sender, System.Windows.Input.TextCompositionEventArgs e)

{

if (!char.IsDigit(e.Text, 0) || tbMinute.Text.Length > 2 ||

(tbMinute.Text.Length > 0 && Convert.ToInt32(tbMinute.Text) >= 59))

e.Handled = true;

}

private void TbSearch\_KeyDown(object sender, System.Windows.Input.KeyEventArgs e)

{

if (e.Key == System.Windows.Input.Key.Enter)

LoadLotsWithFilters();

}

private void TextBox\_TextChanged(object sender, TextChangedEventArgs e)

{

isChanged = true;

}

private void DpAuctionDate\_SelectedDateChanged(object sender, SelectionChangedEventArgs e)

{

isChanged = true;

}

private void Grid\_Loaded(object sender, RoutedEventArgs e)

{

if (aucNo != "")

LoadLotsWithFilters();

}

private void BtnGetCatalog\_Click(object sender, RoutedEventArgs e)

{

DataTable dataAllLots = new DataTable();

DB.LoadDataTable(dataAllLots,

"select lot\_no,lot\_name,lot\_describe,start\_price,id\_lot from lots where auction\_no=" + aucNo);

Word.\_Application wordApp = new Word.Application();

Word.Document doc = wordApp.Documents.Add(Missing.Value, Missing.Value, Missing.Value, Missing.Value);

wordApp.Visible = true;

int x = 1;

for (int i = 1; i <= dataAllLots.Rows.Count; i++)

{

List<byte[]> dataImages = new List<byte[]>();

DB.LoadImageList(dataImages,

"select top 1 \* from images where id\_lot=" + dataAllLots.Rows[i - 1][4]);

doc.Paragraphs.Add();

if (dataImages.Count == 1)

{

Clipboard.SetImage(DB.LoadImage(dataImages[0]));

doc.Paragraphs[x].Range.Paste();

}

doc.Paragraphs.Add();

doc.Paragraphs[x + 1].Range.Text = "Лот №" + dataAllLots.Rows[i - 1][0].ToString();

doc.Paragraphs[x + 1].Range.Bold = 1;

doc.Paragraphs[x + 1].Range.Font.Name = "Lucida Sans Unicode";

doc.Paragraphs[x + 1].Range.Font.Size = 16;

doc.Paragraphs[x + 1].Range.ParagraphFormat.Alignment =

Word.WdParagraphAlignment.wdAlignParagraphCenter;

doc.Paragraphs.Add();

doc.Paragraphs[x + 2].Range.Text = dataAllLots.Rows[i - 1][1].ToString();

doc.Paragraphs[x + 2].Range.Bold = 1;

doc.Paragraphs[x + 2].Range.Font.Name = "Lucida Sans Unicode";

doc.Paragraphs[x + 2].Range.Font.Size = 12;

doc.Paragraphs[x + 2].Range.ParagraphFormat.Alignment =

Word.WdParagraphAlignment.wdAlignParagraphCenter;

doc.Paragraphs.Add();

doc.Paragraphs[x + 3].Range.Bold = 0;

doc.Paragraphs[x + 3].Range.Text = dataAllLots.Rows[i - 1][2].ToString();

doc.Paragraphs[x + 3].Range.Font.Name = "Lucida Sans Unicode";

doc.Paragraphs[x + 3].Range.Font.Size = 10;

doc.Paragraphs[x + 3].Range.ParagraphFormat.Alignment =

Word.WdParagraphAlignment.wdAlignParagraphLeft;

doc.Paragraphs.Add();

doc.Paragraphs[x + 4].Range.Bold = 1;

doc.Paragraphs[x + 4].Range.Text = dataAllLots.Rows[i - 1][3].ToString();

doc.Paragraphs[x + 4].Range.Font.Name = "Lucida Sans Unicode";

doc.Paragraphs[x + 4].Range.Font.Size = 15;

doc.Paragraphs[x + 4].Range.ParagraphFormat.Alignment =

Word.WdParagraphAlignment.wdAlignParagraphLeft;

doc.Paragraphs[x + 4].Range.Text = "Стартовая цена: " + dataAllLots.Rows[i - 1][3].ToString();

doc.Paragraphs.Add();

Word.InlineShape line = doc.Paragraphs[x + 5].Range.InlineShapes.AddHorizontalLineStandard(Missing.Value);

line.Height = 2;

line.Fill.Solid();

line.HorizontalLineFormat.NoShade = true;

line.Fill.ForeColor.RGB = System.Drawing.ColorTranslator.ToOle(System.Drawing.Color.Black);

line.HorizontalLineFormat.PercentWidth = 90;

line.HorizontalLineFormat.Alignment = Word.WdHorizontalLineAlignment.wdHorizontalLineAlignCenter;

x += 7;

}

}

private void BtnGetBids\_Click(object sender, RoutedEventArgs e)

{

DataTable dataBids = new DataTable();

DB.LoadDataTable(dataBids, "select lot\_no,max(price),familia,imya " +

"from bids right join users on bids.bidder\_id = users.id\_user " +

"right join lots on lots.id\_lot = bids.id\_lot " +

"where price in (select max(price) as maxprice from bids " +

"right join lots on bids.id\_lot = lots.id\_lot " +

"group by lot\_no, auction\_no having auction\_no = " + aucNo + ") " +

"group by lot\_no,bidder\_id,familia,imya");

int RowCount = countRows;

int ColumnCount = 4;

int r = 0;

int h = 0;

Word.Document oDoc = new Word.Document();

oDoc.Application.Visible = true;

Word.Range range = oDoc.Content.Application.Selection.Range;

Word.Table table = oDoc.Tables.Add(range, RowCount, ColumnCount, Missing.Value, Missing.Value);

table.Cell(1, 1).Range.Text = "№ лота";

table.Cell(1, 2).Range.Text = "Ставка";

table.Cell(1, 3).Range.Text = "ФИО";

table.Cell(1, 4).Range.Text = "№ участника";

int id = 1;

for (r = 0, h = 2; r < dataBids.Rows.Count & h <= RowCount; h++)

{

if ((int)dataBids.Rows[r][0] != id)

table.Cell(h, 1).Range.Text = id.ToString();

else

{

table.Cell(h, 1).Range.Text = dataBids.Rows[r][0].ToString();

table.Cell(h, 2).Range.Text = dataBids.Rows[r][1].ToString();

table.Cell(h, 3).Range.Text = dataBids.Rows[r][2] + " " + dataBids.Rows[r][3];

r++;

}

id++;

}

table.Borders.OutsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;

table.Borders.InsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;

for (int c = 0; c <= ColumnCount; c++)

{

table.Cell(1, c).Range.Bold = 1;

}

}

public void AddButtons()

{

spPages.Children.Clear();

for (int i = 0; i < countPages; i++)

{

Button bPage = new Button();

bPage.Width = 30;

bPage.Click += new RoutedEventHandler(BtnPage\_Click);

bPage.Content = (i + 1).ToString();

if (i == selectedPage - 1)

bPage.FontWeight = FontWeights.Bold;

spPages.Children.Add(bPage);

}

}

public void LoadLotsWithFilters()

{

lvLotList.ItemsSource = null;

dataAuctionCatalog.Rows.Clear();

lvLotList.Items.Clear();

lotControls.Clear();

string sql = "select \* from lots where auction\_no=" + aucNo;

string stSearch = "";

string stOrder = " order by lot\_no";

if (tbSearch.Text != "")

{

if (int.TryParse(tbSearch.Text, out int number))

stSearch = string.Format(" and lot\_no ={0}", tbSearch.Text);

else

stSearch = string.Format(" and lot\_name like '%{0}%'", tbSearch.Text);

}

switch (cbSort.SelectedIndex)

{

case 0:

stOrder = " order by lot\_no";

break;

case 1:

stOrder = " order by lot\_no desc";

break;

case 2:

stOrder = " order by start\_price";

break;

case 3:

stOrder = " order by start\_price desc";

break;

}

DB.LoadDataTable(dataAuctionCatalog, sql + stSearch + stOrder +

" offset 5\*" + (selectedPage - 1) + " rows fetch next 5 rows only");

countSelected = (int)DB.QueryScalar(

"select count(\*) from lots where auction\_no=" + aucNo + stSearch);

countPages = (int)Math.Ceiling((double)countSelected / 5);

foreach (DataRow row in dataAuctionCatalog.Rows)

{

lotControls.Add(new LotControl(row));

}

lvLotList.ItemsSource = lotControls;

AddButtons();

}

}

}

Форма LotPage.xaml.cs (Лот)

using Auction.Classes;

using Microsoft.Win32;

using System;

using System.Collections.Generic;

using System.Data;

using System.IO;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Input;

using System.Windows.Media;

namespace Auction.Views

{

/// <summary>

/// Логика взаимодействия для LotPage.xaml

/// </summary>

public partial class LotPage : Page

{

List<byte[]> imgList = new List<byte[]>();

List<byte[]> imgAddList = new List<byte[]>();

List<int> imgDelList = new List<int>();

byte[] imageData;

double maxBid;

int selectedImage = 0;

int lotId = -1;

int lotNo = -1;

bool bids = false;

bool isChanged;

public LotPage(int id)

{

InitializeComponent();

if (!SignInPage.isAdmin)

{

btnAddImage.Visibility = Visibility.Collapsed;

btnDeleteImage.Visibility = Visibility.Collapsed;

btnSaveLot.Visibility = Visibility.Collapsed;

btnDeleteLot.Visibility = Visibility.Collapsed;

btnLoadBids.Visibility = Visibility.Hidden;

tStatus.Visibility = Visibility.Collapsed;

rbSold.Visibility = Visibility.Collapsed;

rbNotSold.Visibility = Visibility.Collapsed;

rbWaitSale.Visibility = Visibility.Collapsed;

tbLotName.IsReadOnly = true;

tbLotName.IsReadOnlyCaretVisible = false;

tbLotName.BorderThickness = new Thickness(0);

tbLotName.Background = new SolidColorBrush(Colors.Transparent);

tbStartPrice.IsReadOnly = true;

tbStartPrice.IsReadOnlyCaretVisible = false;

tbStartPric e.BorderThickness = new Thickness(0);

tbStartPrice.Background = new SolidColorBrush(Colors.Transparent);

tbFinalPrice.IsReadOnly = true;

tbFinalPrice.IsReadOnlyCaretVisible = false;

tbFinalPrice.BorderThickness = new Thickness(0);

tbFinalPrice.Background = new SolidColorBrush(Colors.Transparent);

tbLotDescribe.IsReadOnly = true;

tbLotDescribe.IsReadOnlyCaretVisible = false;

tbLotDescribe.BorderThickness = new Thickness(0);

tbLotDescribe.Background = new SolidColorBrush(Colors.Transparent);

tbMaxBid.IsReadOnly = true;

tbMaxBid.IsReadOnlyCaretVisible = false;

tbMaxBid.BorderThickness = new Thickness(0);

tbMaxBid.Background = new SolidColorBrush(Colors.Transparent);

}

if (SignInPage.isAdmin)

{

textBid.Visibility = Visibility.Collapsed;

tbBid.Visibility = Visibility.Collapsed;

btnAddBid.Visibility = Visibility.Collapsed;

}

for (int i = 0; i < AuctionCatalogPage.countRows; i++)

{

cbLotNo.Items.Add(i + 1);

}

DataTable dataLot = new DataTable();

DB.LoadDataTable(dataLot, "select \* from lots where id\_lot=" + id);

LoadLot(dataLot.Rows[0]);

isChanged = false;

}

public LotPage()

{

InitializeComponent();

btnLoadBids.IsEnabled = false;

btnDeleteLot.IsEnabled = false;

tbMaxBid.Visibility = Visibility.Collapsed;

tMaxBid.Visibility = Visibility.Collapsed;

tPrice.Visibility = Visibility.Collapsed;

tbFinalPrice.Visibility = Visibility.Collapsed;

tbBid.Visibility = Visibility.Collapsed;

textBid.Visibility = Visibility.Collapsed;

btnAddBid.Visibility = Visibility.Hidden;

}

private void BtnImgDown\_Click(object sender, RoutedEventArgs e)

{

if (selectedImage - 1 >= 0)

{

selectedImage--;

imgLot.Source = DB.LoadImage(imgList[selectedImage]);

}

else if (selectedImage - 1 < 0)

{

selectedImage = imgList.Count - 1;

imgLot.Source = DB.LoadImage(imgList[selectedImage]);

}

}

private void BtnImgUp\_Click(object sender, RoutedEventArgs e)

{

if (selectedImage + 1 < imgList.Count)

{

selectedImage++;

imgLot.Source = DB.LoadImage(imgList[selectedImage]);

}

else if (selectedImage + 1 == imgList.Count)

{

selectedImage = 0;

imgLot.Source = DB.LoadImage(imgList[selectedImage]);

}

}

private void BtnBackToCatalog\_Click(object sender, RoutedEventArgs e)

{

if (isChanged)

{

MessageBoxResult confirm = MessageBox.Show(

"Вы уверены что хотите выйти? Все несохраненные данные будут утеряны",

"Подтверждение", MessageBoxButton.YesNo, MessageBoxImage.Question);

if (confirm == MessageBoxResult.Yes)

Manager.MainFrame.GoBack();

}

else

Manager.MainFrame.GoBack();

}

private void BtnAddImage\_Click(object sender, RoutedEventArgs e)

{

OpenFileDialog fd = new OpenFileDialog();

if (fd.ShowDialog() == true)

{

using (FileStream fs = new FileStream(fd.FileName, FileMode.Open))

{

imageData = new byte[fs.Length];

fs.Read(imageData, 0, imageData.Length);

}

imgAddList.Add(imageData);

imgList.Add(imageData);

//tbCountUnsaved.Text = "Несохраненных изображений:" + imgAddList.Count;

if (imgList.Count == 1)

imgLot.Source = DB.LoadImage(imgList[0]);

else

imgLot.Source = DB.LoadImage(imgAddList[imgAddList.Count - 1]);

}

}

private void BtnDeleteImage\_Click(object sender, RoutedEventArgs e)

{

if (imgList.Count == 0)

MessageBox.Show("В списке нет изображений");

else

{

MessageBoxResult confirm = MessageBox.Show("Удалить изображение?", "Подтверждение",

MessageBoxButton.YesNo, MessageBoxImage.Question);

if (confirm == MessageBoxResult.Yes)

{

object imageId = DB.QueryScalar("select id\_image from images where id\_lot=" + lotId +

" order by id\_image offset " + selectedImage + " rows fetch next 1 rows only");

if (imageId != null)

imgDelList.Add((int)imageId);

else

{

imgAddList.Remove(imgAddList[imgList.Count - selectedImage]);

}

imgList.Remove(imgList[selectedImage]);

if (selectedImage == imgList.Count && imgList.Count != 0)

{

selectedImage--;

imgLot.Source = DB.LoadImage(imgList[selectedImage]);

}

else if (imgList.Count == 0)

imgLot.Source = null;

else

imgLot.Source = DB.LoadImage(imgList[selectedImage]);

MessageBox.Show("Удалено успешно!");

}

}

}

private void BtnLotDown\_Click(object sender, RoutedEventArgs e)

{

if (lotNo - 1 > 0)

{

DataTable predLot = new DataTable();

DB.LoadDataTable(predLot, "select \* from lots where id\_lot=" + (lotNo - 1));

if (predLot.Rows.Count == 0)

LoadLot(predLot.Rows[0]);

}

}

private void BtnLotUp\_Click(object sender, RoutedEventArgs e)

{

if (lotNo + 1 <= AuctionCatalogPage.countRows)

{

DataTable nextLot = new DataTable();

DB.LoadDataTable(nextLot, "select \* from lots where auction\_no=" +

AuctionCatalogPage.aucNo + "and lot\_no=" + (lotNo + 1));

if(nextLot.Rows.Count==0)

LoadLot(nextLot.Rows[0]);

}

}

private void CbLotNo\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

lotNo = cbLotNo.SelectedIndex;

DataTable selectedLot = new DataTable();

DB.LoadDataTable(selectedLot, "select \* from lots where auction\_no=" +

AuctionCatalogPage.aucNo + "and lot\_no=" + (lotNo + 1));

if (selectedLot.Rows.Count == 0)

LoadLot(selectedLot.Rows[0]);

}

private void BtnSaveLot\_Click(object sender, RoutedEventArgs e)

{

string sql;

if (tbLotName.Text == "")

MessageBox.Show("Введите название лота");

else if (tbStartPrice.Text == "")

MessageBox.Show("Введите начальную цену лота");

else if (tbLotDescribe.Text == "")

MessageBox.Show("Введите описание лота");

else

{

string status = "";

if (rbWaitSale.IsChecked == true)

status = "Ожидает продажи";

else if (rbNotSold.IsChecked == true)

status = "Не продан";

else if (rbSold.IsChecked == true)

status = "Продан";

if (rbWaitSale.IsChecked == false && rbNotSold.IsChecked == false && rbSold.IsChecked == false)

MessageBox.Show("Укажите статус лота");

else

{

if (DB.QueryScalar("select max(lot\_no) from lots " +

"where auction\_no=" + AuctionCatalogPage.aucNo) == DBNull.Value)

lotNo = 1;

else

lotNo = (int)DB.QueryScalar("select max(lot\_no) from lots " +

"where auction\_no=" + AuctionCatalogPage.aucNo) + 1;

if (lotId == -1)

{

sql = string.Format("insert into lots(lot\_name,lot\_describe," +

"lot\_no,auction\_no,start\_price,lot\_status) values(" +

"'{0}','{1}',{2},{3},{4},'{5}')",

tbLotName.Text, tbLotDescribe.Text, lotNo, AuctionCatalogPage.aucNo,

tbStartPrice.Text.Replace(',', '.'), status);

}

else

{

sql = string.Format("update lots set lot\_name='{0}',lot\_describe='{1}'," +

"start\_price={2},lot\_status='{3}' where id\_lot=" + lotId,

tbLotName.Text, tbLotDescribe.Text, tbStartPrice.Text.Replace(',', '.'), status);

}

DB.QueryData(sql);

if (lotId == -1)

{

sql = string.Format("select id\_lot from lots where lot\_no={0} and" +

" start\_price={1} and lot\_status like '{2}' and auction\_no={3}",

lotNo, tbStartPrice.Text.Replace(',', '.'), status, AuctionCatalogPage.aucNo);

lotId = (int)DB.QueryScalar(sql);

lotNo = (int)DB.QueryScalar("select lot\_no from lots where id\_lot=" + lotId);

}

//Вставка изображений из addlist если он не пустой

if (imgAddList.Count != 0)

{

foreach (byte[] image in imgAddList)

{

sql = "insert into images(id\_lot,lot\_img) values(" + lotId + ",@image)";

DB.QueryImage(image, sql);

//tbCountUnsaved.Text = "";

}

}

if (imgDelList.Count != 0)

{

foreach (int imageId in imgDelList)

{

sql = "delete from images where id\_image=" + imageId;

DB.QueryData(sql);

//tbCountUnsaved.Text = "";

}

}

isChanged = false;

btnDeleteLot.IsEnabled = false;

MessageBox.Show("Лот сохранен успешно!");

cbLotNo.SelectedIndex = lotNo - 1;

btnLoadBids.IsEnabled = true;

btnDeleteLot.IsEnabled = true;

}

}

}

private void BtnDeleteLot\_Click(object sender, RoutedEventArgs e)

{

MessageBoxResult confirm = MessageBox.Show(

"Вы точно хотите удалить лот? После удаления данные нельзя будет восстановить.",

"Подтверждение", MessageBoxButton.YesNo, MessageBoxImage.Question);

if (confirm == MessageBoxResult.Yes)

{

string sql = "delete from lots where id\_lot=" + lotId;

DB.QueryData(sql);

sql = string.Format("update lots set lot\_no=" +

"(select l.lot\_no-1 from lots as l where auction\_no={0}) " +

"where auction\_no={0} and lot\_no>{1}", AuctionCatalogPage.aucNo, lotNo);

DB.QueryData(sql);

MessageBox.Show("Удалено успешно!");

Manager.MainFrame.GoBack();

}

}

private void TbPrice\_PreviewTextInput(object sender, TextCompositionEventArgs e)

{

if (!char.IsDigit(e.Text, 0))

e.Handled = true;

}

private void BtnLoadBids\_Click(object sender, RoutedEventArgs e)

{

Manager.MainFrame.Navigate(new BidsPage(lotId));

}

private void BtnAddBid\_Click(object sender, RoutedEventArgs e)

{

if (tbBid.Text == "")

MessageBox.Show("Введите сумму ставки");

else

{

//если ставки были

if (bids)

{

if (System.Convert.ToDouble(tbBid.Text) <= maxBid)

MessageBox.Show("Ставка должна быть выше предыдущей");

else

{

string sql = string.Format("insert into bids values({0},{1},{2})",

lotId, tbBid.Text, SignInPage.userInfo.Rows[0][0]);

DB.QueryData(sql);

MessageBox.Show("Ставка сохранена!");

}

}

//если ставок не было

else

{

if (System.Convert.ToDouble(tbBid.Text) < maxBid)

MessageBox.Show("Ставка не может быть ниже предыдущей");

else

{

string sql = string.Format("insert into bids values({0},{1},{2})",

lotId, tbBid.Text, SignInPage.userInfo.Rows[0][0]);

DB.QueryData(sql);

MessageBox.Show("Ставка сохранена!");

}

}

}

}

private void TextBox\_TextChanged(object sender, TextChangedEventArgs e)

{

isChanged = true;

}

private void RbStatus\_Checked(object sender, RoutedEventArgs e)

{

isChanged = true;

if (rbSold.IsChecked == true)//если лот продан показать финальную стоимость

{

tPrice.Visibility = Visibility.Visible;

tbFinalPrice.Visibility = Visibility.Visible;

}

else if (rbNotSold.IsChecked == true)//если лот не продан скрыть финальную стоимость

{

tPrice.Visibility = Visibility.Collapsed;

tbFinalPrice.Visibility = Visibility.Collapsed;

}

else if (rbWaitSale.IsChecked == true)//если лот не продан скрыть финальную стоимость

{

tPrice.Visibility = Visibility.Collapsed;

tbFinalPrice.Visibility = Visibility.Collapsed;

}

}

public void LoadLot(DataRow row)

{

imgList.Clear();

imgAddList.Clear();

imgDelList.Clear();

tPrice.Visibility = Visibility.Visible;

tbFinalPrice.Visibility = Visibility.Visible;

tMaxBid.Visibility = Visibility.Visible;

tbMaxBid.Visibility = Visibility.Visible;

imgLot.Source = null;

lotId = (int)row[0];

tbLotName.Text = row[1].ToString();

tbLotDescribe.Text = row[2].ToString();

lotNo = (int)row[3];

tbStartPrice.Text = row[5].ToString();

if (row[7].ToString() == "Продан")//если лот продан показать финальную стоимость

{

tPrice.Visibility = Visibility.Visible;

tbFinalPrice.Visibility = Visibility.Visible;

tPrice.Text = "Цена:";

tMaxBid.Visibility = Visibility.Collapsed;

tbMaxBid.Visibility = Visibility.Collapsed;

tbFinalPrice.Text = row[6].ToString();

maxBid = Convert.ToDouble(row[6]);

if (SignInPage.isAdmin)

rbSold.IsChecked = true;

}

else if (row[7].ToString() == "Не продан")//если лот не продан скрыть финальную стоимость

{

tPrice.Visibility = Visibility.Collapsed;

tbFinalPrice.Visibility = Visibility.Collapsed;

tMaxBid.Visibility = Visibility.Collapsed;

tbMaxBid.Visibility = Visibility.Collapsed;

if (SignInPage.isAdmin)

rbNotSold.IsChecked = true;

}

else if (row[7].ToString() == "Ожидает продажи")//если лот ожидает продажи найти наибольшую стаку

{

tPrice.Visibility = Visibility.Collapsed;

tbFinalPrice.Visibility = Visibility.Collapsed;

tMaxBid.Visibility = Visibility.Visible;

tbMaxBid.Visibility = Visibility.Visible;

if (SignInPage.isAdmin)

{

rbWaitSale.IsChecked = true;

tMaxBid.Visibility = Visibility.Collapsed;

tbMaxBid.Visibility = Visibility.Collapsed;

}

else

{

tMaxBid.Visibility = Visibility.Visible;

tbMaxBid.Visibility = Visibility.Visible;

object bidValue = DB.QueryScalar("select max(price) from bids where id\_lot=" + lotId);

if (bidValue == DBNull.Value)

maxBid = 0;

else

maxBid = Convert.ToDouble(bidValue);

if (maxBid > 0)//если ставки были поменять значение флажка и вывести на экран наибольшую ставку

{

bids = true;

tbMaxBid.Text = maxBid.ToString();

}

else//если ставок не было запомнить начальную цену и использовать ее как наибольшую ставку

{

bids = false;

tbMaxBid.Text = maxBid.ToString();

maxBid = Convert.ToDouble(row[5]);

}

}

}

if (!AuctionCatalogPage.isFuture)

{

textBid.Visibility = Visibility.Collapsed;

tbBid.Visibility = Visibility.Collapsed;

btnAddBid.Visibility = Visibility.Collapsed;

}

DB.LoadImageList(imgList, "select \* from images where id\_lot=" + row[0]);

if (imgList.Count > 0)

{

selectedImage = 0;

imgLot.Source = DB.LoadImage(imgList[0]);

}

cbLotNo.SelectedIndex = lotNo - 1;

isChanged = false;

}

}

}

Форма BidsPage.xaml.cs (Ставки)

using Auction.Classes;

using System.Collections.Generic;

using System.Data;

using System.Windows;

using System.Windows.Controls;

namespace Auction.Views

{

/// <summary>

/// Логика взаимодействия для BidsPage.xaml

/// </summary>

public partial class BidsPage : Page

{

List<BidControl> bidControls = new List<BidControl>();

public BidsPage()

{

InitializeComponent();

if (!SignInPage.isAdmin)

{

DataTable dataBids = new DataTable();

DB.LoadDataTable(dataBids, "select lots.\*,price,bidder\_id from lots inner join bids on bids.id\_lot = lots.id\_lot where bidder\_id=" + SignInPage.userInfo.Rows[0][0]);

foreach (DataRow row in dataBids.Rows)

{

bidControls.Add(new BidControl(row));

}

lvBids.ItemsSource = bidControls;

}

}

public BidsPage(int idLot)

{

InitializeComponent();

if (SignInPage.isAdmin)

{

DataTable dataBids = new DataTable();

DB.LoadDataTable(dataBids, "select users.\*, price from bids inner join users on bids.bidder\_id = users.id\_user where bids.id\_lot =" + idLot);

foreach (DataRow row in dataBids.Rows)

{

bidControls.Add(new BidControl(row));

}

lvBids.ItemsSource = bidControls;

}

}

private void BtnBack\_Click(object sender, RoutedEventArgs e)

{

Manager.MainFrame.GoBack();

}

}

}

Форма UserInfoPage.xaml.cs (Информация о пользователе)

using Auction.Classes;

using System;

using System.Data;

using System.Windows;

using System.Windows.Controls;

namespace Auction.Views

{

/// <summary>

/// Логика взаимодействия для UserInfoPage.xaml

/// </summary>

public partial class UserInfoPage : Page

{

bool isInsert;

public UserInfoPage()

{

InitializeComponent();

if (SignInPage.userInfo.Rows.Count == 0)

{

tbInfo.Text = "Регистрация";

isInsert = true;

}

else

{

tbFamilia.Text = SignInPage.userInfo.Rows[0][1].ToString();

tbImya.Text = SignInPage.userInfo.Rows[0][2].ToString();

tbOtchestvo.Text = SignInPage.userInfo.Rows[0][3].ToString();

tbAddress.Text = SignInPage.userInfo.Rows[0][5].ToString();

tbEmail.Text = SignInPage.userInfo.Rows[0][6].ToString();

tbPhone.Text = SignInPage.userInfo.Rows[0][7].ToString();

tbLogin.Text = SignInPage.userInfo.Rows[0][8].ToString();

tbPassword.Text = SignInPage.userInfo.Rows[0][9].ToString();

dpBirtday.SelectedDate = DateTime.Parse(SignInPage.userInfo.Rows[0][4].ToString());

isInsert = false;

}

}

public UserInfoPage(string id)

{

InitializeComponent();

DataTable dataUser = new DataTable();

DB.LoadDataTable(dataUser, "select \* from users where id\_user=" + id);

btnSaveUser.Visibility = Visibility.Visible;

tbFamilia.IsReadOnly = true;

tbFamilia.IsReadOnlyCaretVisible = false;

tbFamilia.Text = dataUser.Rows[0][1].ToString();

tbImya.IsReadOnly = true;

tbImya.IsReadOnlyCaretVisible = false;

tbImya.Text = dataUser.Rows[0][2].ToString();

tbOtchestvo.IsReadOnly = true;

tbOtchestvo.IsReadOnlyCaretVisible = false;

tbOtchestvo.Text = dataUser.Rows[0][3].ToString();

tbAddress.IsReadOnly = true;

tbAddress.IsReadOnlyCaretVisible = false;

tbAddress.Text = dataUser.Rows[0][5].ToString();

tbEmail.IsReadOnly = true;

tbEmail.IsReadOnlyCaretVisible = false;

tbEmail.Text = dataUser.Rows[0][6].ToString();

tbPhone.IsReadOnly = true;

tbPhone.IsReadOnlyCaretVisible = false;

tbPhone.Text = dataUser.Rows[0][7].ToString();

dpBirtday.IsEnabled = false;

dpBirtday.SelectedDate = DateTime.Parse(dataUser.Rows[0][4].ToString());

tbLogin.Visibility = Visibility.Collapsed;

textLogin.Visibility = Visibility.Collapsed;

tbPassword.Visibility = Visibility.Collapsed;

textPassword.Visibility = Visibility.Collapsed;

btnSaveUser.Visibility = Visibility.Hidden;

}

private void BtnSaveUser\_Click(object sender, RoutedEventArgs e)

{

if (tbFamilia.Text == "")

MessageBox.Show("Укажите фамилию");

else if (tbImya.Text == "")

MessageBox.Show("Укажите имя");

else if (tbOtchestvo.Text == "")

MessageBox.Show("Укажите отчество");

else if (dpBirtday.Text == "Выбор даты")

MessageBox.Show("Укажите дату рождения");

else if (tbEmail.Text == "")

MessageBox.Show("Укажите адрес электронной почты");

else if (tbPhone.Text == "")

MessageBox.Show("Укажите номер телефона");

else if(tbLogin.Text.Length>15)

MessageBox.Show("Логин не может быть больше 15 символов");

else if (tbPassword.Text.Length > 15)

MessageBox.Show("Пароль не может быть больше 15 символов");

else

{

if (tbAddress.Text == "")

tbAddress.Text = "не указан";

string login="";

string password="";

if (tbLogin.Text == "")

{

if (!isInsert)

login = SignInPage.userInfo.Rows[0][8].ToString();

else

MessageBox.Show("Укажите логин");

}

else

login = tbLogin.Text;

if (tbPassword.Text == "")

{

if (!isInsert)

password = SignInPage.userInfo.Rows[0][9].ToString();

else

MessageBox.Show("Укажите пароль");

}

else

password = tbPassword.Text;

if ((int)DB.QueryScalar("select count(\*) from users where login='" + login+"'") == 0)

{

string sqlFormattedDate = dpBirtday.SelectedDate.Value.ToString("yyyy-dd-MM");

string sql;

if (isInsert == false)

{

sql = string.Format("update users set familia='{0}',imya='{1}',otchestvo='{2}',birthday='{3}'," +

"registr\_adress='{4}',el\_adress='{5}',telephone='{6}',login='{7}',password='{8}' where id\_user={9}", tbFamilia.Text, tbImya.Text,

tbOtchestvo.Text, sqlFormattedDate, tbAddress.Text, tbEmail.Text, tbPhone.Text, login, password, SignInPage.userInfo.Rows[0][0]);

}

else

{

sql = string.Format("insert into users values('{0}','{1}','{2}','{3}','{4}','{5}','{6}','{7}','{8}',{9})",

tbFamilia.Text, tbImya.Text, tbOtchestvo.Text,

sqlFormattedDate, tbAddress.Text, tbEmail.Text, tbPhone.Text, login, password, 0);

}

DB.QueryData(sql);

MessageBox.Show("Данные сохранены!");

}

else

MessageBox.Show("Пользователь с таким логином уже существует!");

}

}

private void BtnBack\_Click(object sender, RoutedEventArgs e)

{

Manager.MainFrame.GoBack();

}

}

}

Форма AuctionControl.xaml.cs (Отображение информации об аукционе)

using Auction.Classes;

using System;

using System.Collections.Generic;

using System.Data;

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.Navigation;

using System.Windows.Shapes;

namespace Auction.Views

{

/// <summary>

/// Логика взаимодействия для AuctionControl.xaml

/// </summary>

public partial class AuctionControl : UserControl

{

DataRow auction;

bool isFuture;

public AuctionControl(DataRow row,bool b)

{

InitializeComponent();

auction = row;

isFuture = b;

tbAuctionName.Text = auction[1].ToString();

if (auction[4].ToString() != "")

{

byte[] imageData = (byte[])auction[4];

imgAuctionCover.Source= DB.LoadImage(imageData);

}

tbDate.Text = auction[5].ToString();

Tag = auction[0];

}

private void Grid\_MouseLeftButtonDown(object sender, MouseButtonEventArgs e)

{

Manager.MainFrame.Navigate(new AuctionCatalogPage(auction,isFuture));

}

}

}

Форма LotControl.xaml.cs (Отображение информации о лоте)

using Auction.Classes;

using System.Data;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Input;

namespace Auction.Views

{

/// <summary>

/// Логика взаимодействия для LotControl.xaml

/// </summary>

public partial class LotControl : UserControl

{

int id;

public LotControl(DataRow row)

{

InitializeComponent();

id = (int)row[0];

tbLotName.Text = row[1].ToString();

tbLotNo.Text = "Лот №" + row[3].ToString();

byte[] imageData = (byte[])DB.QueryScalar("select top 1 lot\_img from images where id\_lot=" + row[0].ToString());

if (imageData != null)

imgLotPrewiew.Source = DB.LoadImage(imageData);

tbPrice.Text = row[5].ToString();

if (row[7].ToString() == "Продан")

{

textFinalPrice.Visibility = Visibility.Visible;

tbFinalPrice.Text = row[6].ToString();

}

}

private void Grid\_MouseLeftButtonDown(object sender, MouseButtonEventArgs e)

{

Manager.MainFrame.Navigate(new LotPage(id));

}

}

}

Форма BidControl.xaml.cs (Отображение информации о ставке)

using Auction.Classes;

using System.Data;

using System.Windows.Controls;

using System.Windows.Input;

namespace Auction.Views

{

/// <summary>

/// Логика взаимодействия для BidControl.xaml

/// </summary>

public partial class BidControl : UserControl

{

DataRow info;

public BidControl(DataRow row)

{

InitializeComponent();

info = row;

if (!SignInPage.isAdmin)

{

tbLotName.Text = info[1].ToString();

tbInfo2.Text = "Лот №" + info[3].ToString();

tbPrice.Text = info[5].ToString();

tbInfo1.Text = "Аукцион №" + info[4].ToString();

textPrice.Text = "Ваша ставка:";

tbPrice.Text = info[8].ToString();

textForAction.Text = "Нажмите для просмотра информации о лоте";

}

else

{

imgLotPrewiew.Visibility = System.Windows.Visibility.Collapsed;

tbInfo1.Text = info[1].ToString();

tbInfo2.Text = info[2].ToString();

textPrice.Text = "Cтавка:";

tbPrice.Text = info[11].ToString();

textForAction.Text = "Нажмите для просмотра информации о пользователе";

}

}

private void Border\_MouseLeftButtonDown(object sender, MouseButtonEventArgs e)

{

if (!SignInPage.isAdmin)

Manager.MainFrame.Navigate(new LotPage((int)info[0]));

else

Manager.MainFrame.Navigate(new UserInfoPage(info[9].ToString()));

}

}

}

Класс Connection.cs (Соединение с базой данных)

using MySql.Data.MySqlClient;

using System;

using System.Collections.Generic;

using System.Data.SqlClient;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Auction.Classes

{

class Connection

{

public static string connString = @"Data source=LAPTOP-07VNMC1R\SQLEXPRESS; Initial Catalog=db\_auction;Integrated Security=true;";

public static SqlConnection connection = new SqlConnection(connString);

public static void OpenConnection()

{

if (connection.State == System.Data.ConnectionState.Closed)

connection.Open();

}

public static void CloseConnection()

{

if (connection.State == System.Data.ConnectionState.Open)

connection.Close();

}

}

}

Класс DB.cs (Работа с базой данных)

using System.Collections.Generic;

using System.Data;

using System.Data.SqlClient;

using System.IO;

using System.Windows.Media.Imaging;

namespace Auction.Classes

{

class DB

{

public static void LoadImageList(List<byte[]> images, string sql)

{

Connection.OpenConnection();

SqlCommand command = new SqlCommand(sql, Connection.connection);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

byte[] data = (byte[])reader.GetValue(2);

images.Add(data);

}

Connection.CloseConnection();

}

public static BitmapImage LoadImage(byte[] imageData)

{

if (imageData == null || imageData.Length == 0) return null;

var image = new BitmapImage();

using (var mem = new MemoryStream(imageData))

{

mem.Position = 0;

image.BeginInit();

image.CreateOptions = BitmapCreateOptions.PreservePixelFormat;

image.CacheOption = BitmapCacheOption.OnLoad;

image.UriSource = null;

image.StreamSource = mem;

image.EndInit();

}

image.Freeze();

return image;

}

public static void LoadDataTable(DataTable dt, string sql)

{

Connection.OpenConnection();

SqlDataAdapter adapter = new SqlDataAdapter(sql, Connection.connection);

adapter.Fill(dt);

Connection.CloseConnection();

}

public static object QueryScalar(string sql)

{

object sqlValue = null;

Connection.OpenConnection();

try

{

SqlCommand command = new SqlCommand(sql, Connection.connection);

sqlValue = command.ExecuteScalar();

}

catch { }

Connection.CloseConnection();

return sqlValue;

}

public static void QueryData(string sql)

{

Connection.OpenConnection();

try

{

SqlCommand command = new SqlCommand(sql, Connection.connection);

command.ExecuteNonQuery();

}

catch { }

Connection.CloseConnection();

}

public static void QueryImage(byte[] image, string sql)

{

Connection.OpenConnection();

try

{

SqlCommand command = new SqlCommand(sql, Connection.connection);

command.Parameters.Add("@image", SqlDbType.Image, 1000000);

command.Parameters["@image"].Value = image;

command.ExecuteNonQuery();

}

catch { }

Connection.CloseConnection();

}

}

}

Класс Manager.cs (Навигация по страницам)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Controls;

namespace Auction.Classes

{

class Manager

{

public static Frame MainFrame { get; set; }

}

}

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Потопахин В. Искусство алгоритмизации. [Текст] / В. Потопахин. – Москва: ДМК Пресс, 2018. – 320 с.
2. Семакин И. Основы алгоритмизации и программирования. Учебник [Текст] / И. Семакин, А. Шестаков. – 2-е изд., Москва: Академия, 2018. – 304 с.
3. Скиена С. Алгоритмы. Руководство по разработке [Текст] / С. Скиена 2-е изд. – СПб. БХВ-Петербург, 2016. – 720 с.
4. Шарп, Дж. Microsoft Visual C#. Подробное руководство. [Текст]: учеб. / Дж. Шарп – 8-е изд., СПб.: Питер СПб, 2017 – 848 с.
5. Программирование на C, C# и Java [Электронный ресурс] / MySQL и C# работаем с базой данных. – Режим доступа https://vscode.ru/proglessons/mysql-c-sharp.html, свободный
6. Metanit [Электронный ресурс] / Руководство по языку C#. – Режим доступа https://metanit.com/sharp/tutorial/, свободный
7. Metanit [Электронный ресурс] / Руководство по WPF. – Режим доступа https://metanit.com/sharp/wpf/, свободный
8. Microsoft [Электронный ресурс] / Руководство по языку C#. – Режим доступа https://docs.microsoft.com/ru-ru/dotnet/csharp/, свободный
9. Professor Web [Электронный ресурс] / Основы WPF. – Режим доступа https://professorweb.ru/my/WPF/base\_WPF/level1/info\_WPF.php, свободный
10. Ruseller [Электронный ресурс] / Введение в хранимые процедуры MySQL 5. – Режим доступа https://ruseller.com/lessons.php?id=1189, свободный
11. Sql-tutorial [Электронный ресурс] / SQL Задачи и решения. – Режим доступа http://www.sql-tutorial.ru/ru/content.html, свободный
12. W3schools [Электронный ресурс] / SQL Tutorial. – Режим доступа https://www.w3schools.com/sql/default.asp, свободный
13. Webmasterwiki [Электронный ресурс] / Базы данных: Основные понятия. – Режим доступа https://www.webmasterwiki.ru/MySQL, свободный
14. WorldSkills Russia [Электронный ресурс] / Программные решения для бизнеса. – Режим доступа https://nationalteam.worldskills.ru/skills/programmnye-resheniya-dlyabiznesa/, свободны

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. [ГОСТ 28195-89](https://gost.ruscable.ru/Index/11/11212.htm) Оценка качества программных средств. Общие положения: межгосударственный стандарт: дата введения 1990-07-01 - <https://docs.cntd.ru/document/1200009135?ysclid=lroox3amxp882669222> – Текст: электронный.
2. [ГОСТ 28806-90](https://gost.ruscable.ru/Index/10/10605.htm) Качество программных средств. Термины и определения: межгосударственный стандарт: дата введения 1992-01-01 - <https://docs.cntd.ru/document/1200009077?ysclid=lrq6to4u1z728775522> – Текст: электронный.
3. [ГОСТ 34.321-96](https://gost.ruscable.ru/Index/6/6808.htm) Информационные технологии. Система стандартов по базам данных. Эталонная модель управления данными: межгосударственный стандарт: дата введения 2001-07-01 - <https://docs.cntd.ru/document/1200017662?ysclid=lrq6wn65z5753511468> – Текст: электронный.
4. [ГОСТ Р ИСО/МЭК 12207-99. Информационная технология. Процессы жизненного цикла программных средств](https://docs.cntd.ru/document/1200009075#7D20K3): межгосударственный стандарт: дата введения 2000-07-01 -https://docs.cntd.ru/document/1200009075?ysclid=lrool2t1vo124212660. – Текст: электронный.
5. Ботрос, С. MySQL по максимуму / C. Ботрос, Д. Тинли., – 4-е изд., перераб. и доп. – Питер, 2023. – 432 с. – ISBN 978-5-4461-2261-5. – Текст: непосредственный.
6. Доусон, М. Программируем на Python / М. Доусон., перераб. и доп. – Питер, 2022. – 416 с. – ISBN 978-5-4461-1386-6. – Текст: непосредственный.
7. Кириченко, А.В. Web на практике. CSS, HTML, JavaScript, MySQL, PHP для fullstack-разработчиков / А.В. Кириченко, А.П. Никольский, Е.В. Дубовик., перераб. и доп. – Питер, 2021. – 432 с. – ISBN 978-5-94-387271-6. – Текст: непосредственный.
8. Макеев, Г.А. Объектно-ориентированное программирование: с нуля к SOLID и MVC / Г.А. Макеев., перераб. и доп. – БХВ, 2024. – 272 с. – ISBN 978-5-9775-1913-7. – Текст: непосредственный.
9. Постолит, А.В. Python, Django и Bootstrap для начинающих / А.В. Постолит., переаб. и доп. – БХВ, 2023. – 624 с. – ISBN 978-5-9775-1807-9. – Текст: непосредственный
10. Стефанов, С. React. Быстрый старт / С. Стефанов., – 2-е изд., перераб. и доп. – Питер, 2023. – 304 с. – ISBN 978-5-4461-2115-1. – Текст: непосредственный.
11. Тараканов, О.В. Базы данных. Учебник. Студентам ССУЗов / О.В. Тараканов, Л.И. Шустова., 2023. – 304 с. – ISBN 978-5-16-010485-0. - Текст: непосредственный.
12. Трэйси, О. Веб-дизайн для недизайнеров / О. Трэйси., перераб. и доп. – Питер, 2022. – 176 с. – ISBN 978-5-4461-1917-2. – Текст: непосредственный.
13. Документация CSS. – 2024. Текст : электронный. – URL: <https://developer.mozilla.org/ru/docs/Web/CSS/Reference> (дата обращения: 26.01.2024).
14. Документация Django. – 2021. Текст : электронный. – URL: <https://djangodoc.ru/3.2/> (дата обращения: 26.01.2024).
15. Документация HTML. – 2024. Текст : электронный. – URL: <https://developer.mozilla.org/ru/docs/Web/HTML> (дата обращения: 26.01.2024).
16. Документация JavaScript. – 2024. Текст : электронный. – URL: <https://developer.mozilla.org/ru/docs/Web/JavaScript> (дата обращения: 26.01.2024).
17. Документация MySQL. – 2024. Текст : электронный. – URL: <https://dev.mysql.com/doc/> (дата обращения: 26.01.2024).
18. Документация Python. – 2024. Текст : электронный. – URL: <https://docs.python.org/3/index.html> (дата обращения: 26.01.2024).
19. Документация React. – 2023. Текст : электронный. – URL: <https://react.dev/learn> (дата обращения: 26.01.2024).
20. Документация Яндекс Карты API. Текст : электронный. – URL: <https://yandex.ru/maps-api/docs?ysclid=lrq7c8dsns226257630> (дата обращения: 26.01.2024).

1. PK-первичный ключ

   FK-внешний ключ [↑](#footnote-ref-1)