**ГАПОУ «Казанский авиационно-технический колледж имени**

**П.В. Дементьева»**

|  |
| --- |
| **ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**  **к курсовому проекту** |
| **КП 09.02.04.20И1.21.000** |
| (обозначение документа)  **МДК.02.04 Программные решения для бизнеса** |
| Тема: Разработка информационной системы на основе СУБД по теме:  «Конкурс творческих профессий» |

2023

Содержание

[Введение 4](#_Toc151131155)

[1 Анализ предметной области 5](#_Toc151131156)

[1.1 Описание деятельности компании 5](#_Toc151131157)

[1.2 Описание основного бизнес-процесса 6](#_Toc151131158)

[1.3 Определение пользователей и обеспечение безопасности приложения 6](#_Toc151131159)

[1.4 Диаграмма прецедентов 7](#_Toc151131160)

[2. Проектирование базы данных 8](#_Toc151131161)

[2.1 Создание схемы данных 8](#_Toc151131162)

[3. Разработка базы данных 9](#_Toc151131163)

[3.1 Создание словаря данных 9](#_Toc151131164)

[3.2 Создание базы данных 11](#_Toc151131165)

[3.3 Импорт данных 12](#_Toc151131166)

[4. Разработка приложения 16](#_Toc151131167)

[4.1 Разработка стиля 16](#_Toc151131168)

[4.2 Разработка главного окна системы 17](#_Toc151131169)

[4.3 Разработка системы авторизации 19](#_Toc151131170)

[4.4 Разработка окон для работы пользователей 19](#_Toc151131171)

[5. Разработка руководства пользователя 22](#_Toc151131172)

[Заключение 34](#_Toc151131173)

[Информационные источники 35](#_Toc151131174)

[Приложение A 36](#_Toc151131175)

[Приложение Б 38](#_Toc151131176)

[Приложение В 43](#_Toc151131177)

[Приложение Г 48](#_Toc151131178)

# Введение

В современном мире, где музыкальные конкурсы играют ключевую роль в развитии талантов и культурной индустрии, эффективное управление и организация данных о участниках, жюри и мероприятиях становятся все более важными задачами для компаний, проводящих подобные события. Музыкальные конкурсы требуют точной и структурированной информации для гарантированного успеха и удовлетворения всех участвующих в проведении музыкального конкурса.

Однако, часто организации, проводящие музыкальные конкурсы, могут сталкиваться с проблемами в управлении информацией, связанной с этими мероприятиями. Ручное ведение данных и несистематизированные процессы могут привести к ошибкам, потере времени и недовольству участников.

Для решения этих проблем и повышения эффективности проведения музыкальных конкурсов, проект направлен на разработку автоматизированной информационной системы (АИС). Эта система станет мощным инструментом, который поможет улучшить процессы управления, минимизировать ошибки и обеспечить высокое качество мероприятий.

В рамках курсового проекта мы будем выполнять следующие задачи:

1. Анализ предметной области

2. Проектирование и разработка базы данных

3. Разработка приложения

# 1 Анализ предметной области

## 1.1 Описание деятельности компании

"ООО Музыкальное искусство" - это компания, специализирующаяся на организации и проведении музыкальных конкурсов и событий в области музыки. Компания была основана с целью поддерживать и развивать музыкальное искусство, предоставляя музыкантам и исполнителям платформу для проявления своих талантов.

Компания специализируется на организации музыкальных конкурсов разного масштаба и направленности, а также обладает многолетним опытом в организации музыкальных мероприятий и конкурсов и имеет хорошую репутацию в музыкальной индустрии.

В "ООО Музыкальное искусство " работают профессионалы в области организации мероприятий, музыкальные эксперты и технические специалисты. Каждый из них вкладывает свой опыт и знания в создание успешных конкурсов.

Перед компанией поставлены следующие задачи:

1. Планирование мероприятия - определение даты, места проведения и разработка расписания
2. Реклама и продвижение - привлечение участников и зрителей
3. Подготовка музыкальных площадок - обеспечение качественного звукового и светового оборудования на месте проведения конкурса.
4. Подбор жюри и модераторов
5. Проведение конкурса - это ядро мероприятия. Оно включает регистрацию мероприятия в информационной системе, регистрацию участников, голосование и определение победителя
6. Проведение церемонии награждения по итогам конкурса
7. Правовые аспекты - обеспечение соблюдения всех правовых норм и лицензий при проведении конкурса

## 1.2 Описание основного бизнес-процесса

Основные бизнес-процессы:

1. Планирование мероприятия
2. Реклама и продвижение конкурса
3. Подготовка музыкальных площадок
4. Подбор жюри, модераторов и участников
5. Проведение конкурса
6. Проведение церемонии награждения
7. Управление бюджетом

Информационная система направлена на автоматизацию проведения конкурса, а именно следующих задач:

1. Облегчение работы с данными конкурсов

2. Валидация данных

3. Автоматизация прикрепления участников, модераторов и жюри к конкурсу

4. Автоматизация определения победителей конкурса

5. Хранение информации о проводимых конкурсах.

## 1.3 Определение пользователей и обеспечение безопасности приложения

Пользователи системы:

1. Гости
2. Участники
3. Жюри
4. Руководители
5. Организаторы

Гость может просматривать информацию о участниках конкурса и его результатах.

Участник наследует функционал гостя, а также может авторизоваться и изменять личные данные.

Жюри наследует функционал участника и имеет возможность проголосовать по своему направлению за выбранного участника.

Руководитель наследует функционал участника, а также имеет доступ к проверке присутствующих участников и их дисквалификации. Кроме того руководитель оканчивает проведения направления конкурса.

Организатор наследует функционал участника и работает с данными конкурсов, жюри, руководителей и направлений – добавляет, удаляет и редактирует.

Все пользователи системы, кроме гостя, имеют доступ к авторизации. Для авторизации необходимо ввести логин и пароль, созданный для пользователей организаторами конкурса. Пароль должен содержать заглавные буквы, спец.символы и цифры, а также быть не короче 6 символов. Логин должен отражать статус пользователя.

## 1.4 Диаграмма прецедентов

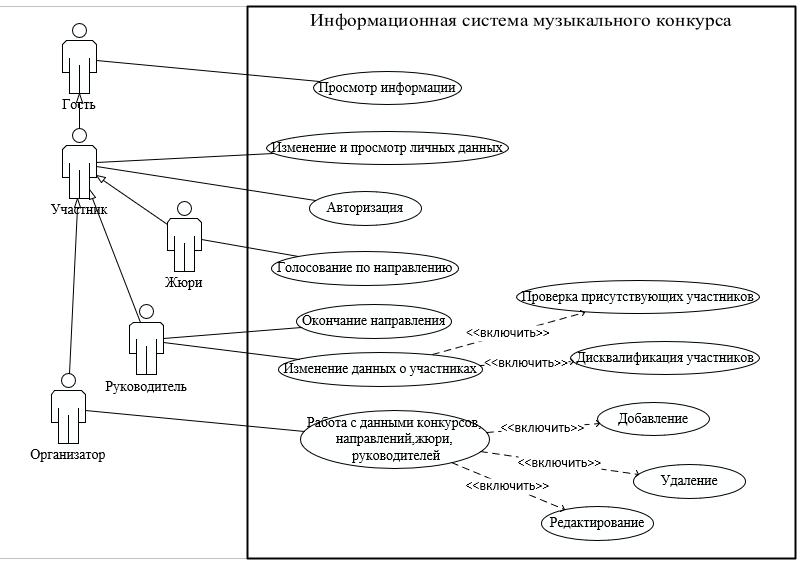


Рисунок 1 – диаграмма прецедентов

# 2. Проектирование базы данных

## 2.1 Создание схемы данных

Были выделены следующие основные сущности: пользователи, конкурсы, направления. На основе данных сущностей была разработана схема данных, представленная в графических материалах

# 3. Разработка базы данных

## 3.1 Создание словаря данных

Для подробного описания таблиц базы данных на основе схемы данных был создан словарь данных, представленный на таблице 1.

Таблица 1 – Словарь данных

|  |  |  |
| --- | --- | --- |
| Direction | | |
| Поле | Тип данных | Примечание |
| direction\_id | int | PK |
| name | nvarchar(30) |  |
| start\_date | date |  |
| start\_time | time(7) |  |
| end\_time | time(7) |  |
| User | | |
| Поле | Тип данных | Примечание |
| user\_id | int | PK |
| f\_name | nvarchar(30) |  |
| l\_name | nvarchar(30) |  |
| patronymic | nvarchar(30) |  |
| user\_type\_id | int | FK UserType.  user\_type\_id |
| photo\_name | nvarchar(50) |  |
| phone | nvarchar(20) |  |
| email | nvarchar(50) |  |
| login | nvarchar(50) |  |
| password | nvarchar(50) |  |
| Voting | | |
| Поле | Тип данных | Примечание |
| voting\_id | int | PK |
| jury | int | FK Concourse\_user.concourse\_user\_id |
| participant | int | FK Concourse\_user.concourse\_user\_id |
| Concourse | | |
| Поле | Тип данных | Примечание |
| concourse\_id | int | PK |
| name | nvarchar(50) |  |
| start\_date | date |  |
| end\_date | date |  |

Окончание таблицы 1

|  |  |  |
| --- | --- | --- |
| Concourse\_Direction | | |
| Поле | Тип данных | Примечание |
| concourse\_direction\_id | int | PK |
| concourse | int | FK Concourse.  concourse\_id |
| direction | int | FK Direction.  direction\_id |
| moderator | int | FK Concourse\_User.concourse  \_user\_id |
| winner |  | FK Concourse\_User.concourse  \_user\_id |
| is\_ended | bit |  |
| start\_date | date |  |
| start\_time | datetime |  |
| end\_time | datetime |  |
| ParticipantDisqualification | | |
| Поле | Тип данных | Примечание |
| participant | int | FK Concourse\_User.concourse  \_user\_id |
| motive | nvarchar(100) |  |
| disqualification\_id | int | PK |
| UserType | | |
| Поле | Тип данных | Примечание |
| user\_type\_id | int | PK |
| name | nvarchar(30) |  |
| ConcourseUsers | | |
| Поле | Тип данных | Примечание |
| concourse\_direction\_id | int | FK Concourse\_Direction.  concourse\_direction\_id |
| concourse\_user\_id | int | PK |
| user\_id | int | FK User.user\_id |
| is\_performed | bit |  |
| song | nvarchar(50) |  |
| golosCoint | int |  |

## 3.2 Создание базы данных

Запустил SQL Server Managment Studio. Подключился к серверу "tenebrepc". Выбрал базы данных, создать базу данных. Ввел название базы данных – "concourse". Нажал ОК.

Создание таблицы User

Раскрыл созданную базу данных. Выбрал папку таблицы. Нажал создать таблицу. Ввел название поля userId. Задал тип данных int. Нажал ПКМ по полю, выбрал первичный ключ. Ввел название поля userType. Выбрал тип данных int. Ввел название поля f\_name. Выбрал тип данных nvarchar(30),снял флажок разрешить значения NULL. Ввел название поля l\_name. Выбрал тип данных nvarchar(30), снял флажок разрешить значения NULL. Ввел название поля patronymic. Выбрал тип данных nvarchar(30), снял флажок разрешить значения NULL. Ввел название поля photo\_name. Выбрал тип данных nvarchar(50) Ввел название поля phone. Выбрал тип данных nvarchar(20), снял флажок разрешить значения NULL. Ввел название поля email. Выбрал тип данных nvarchar(50), снял флажок разрешить значения NULL. Ввел название поля login. Выбрал тип данных nvarchar(50), снял флажок разрешить значения NULL. Ввел название поля password. Выбрал тип данных nvarchar(50), снял флажок разрешить значения NULL. Сохранил таблицу под названием User.

Раскрыл созданную таблицу. Выбрал папку ограничения. Нажал создать ограничение. Выбрал имя ограничения CheckPassword. В поле выражение написал: (len([password])>=(6) AND [password] like '%[a-z]%' AND [password] like '%[A-Z]%' AND [password] like '%[0-9]%' AND [password] like '%[!@#%^&\*()]%').

Аналогичным образом добавил ограничение для проверки телефона.

Раскрыл созданную таблицу. Выбрал папку ключи. Нажал "Создать внешний ключ". В поле имя ввел FK\_User\_UserType. В поле "Спецификация таблиц и столбцов" выбрал в качестве таблицы внешнего ключа UserType и поле user\_type\_id, для таблицы User выбрал поле user\_type\_id.

Создание таблицы Concourse

Открыл существующую базу данных. Выбрал папку для таблицы. Нажал "Создать таблицу". Ввел название поля concourse\_id. Выбрал тип данных int. Нажал правой кнопкой мыши по полю и выбрал "Первичный ключ". Ввел название поля name. Выбрал тип данных nvarchar(50), снял флажок "Разрешить значения NULL". Ввел название поля start\_date. Выбрал тип данных date, снял флажок "Разрешить значения NULL". Ввел название поля end\_date. Выбрал тип данных date, снял флажок "Разрешить значения NULL". Ввел название поля is\_ended. Выбрал тип данных bit, установил значение по умолчанию ((0)), снял флажок "Разрешить значения NULL". Сохранил таблицу под названием Concourse.

Открыл созданную таблицу. Выбрал папку "Ограничения". Нажал "Создать ограничение". Выбрал имя ограничения CK\_Concourse. В поле "Выражение" написал: ([start\_date]<[end\_date]).

Аналогичным образом создал остальные таблицы.

Скрипт по созданию базы данных представлен в приложении A.

## 3.3 Импорт данных

Подготовил данные для импорта в файле Данные.xlsx согласно схеме данных (см. рисунок 2)

Выделил базу данных ПКМ, выбрал задачи, импорт данных.

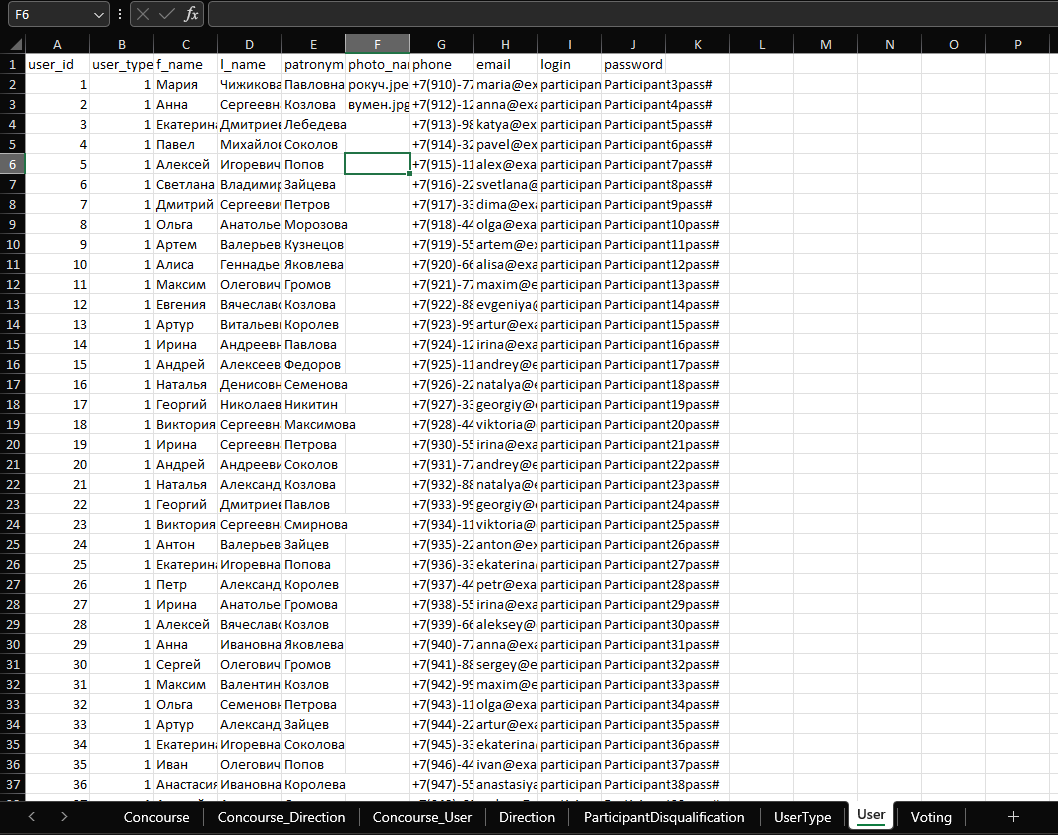
В всплывающем окне в качестве источника данных выбрал Microsoft Excel. Указал путь к файлу "Данные.xlsx". Нажал далее.

В новом окне в поле назначение выбрал Microsoft OLE DB Provider for SQL Server. Указал имя сервера "tenebrepc" и базу данных "concourse". Нажал далее.

В окне "Выбор копирования таблицы или запроса" выбрал скопировать данные из одной или нескольких таблиц или представлений. Нажал далее.

В окне "Выбор исходных таблиц и представлений" сопоставил листы Excel с таблицами БД. Импорт данных успешно выполнен (см. рисунки 3,4,5,6)

Скрипт базы данных предоставлен в приложении A.

Рисунок 2 – Данные для импорта

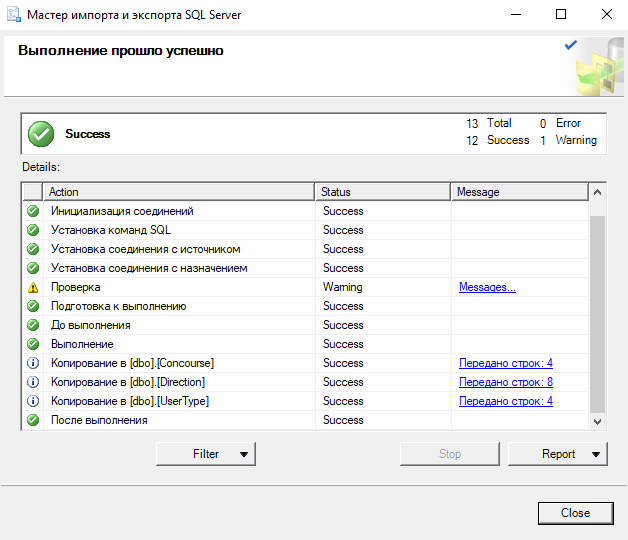


Рисунок 3 – Импорт, шаг 1

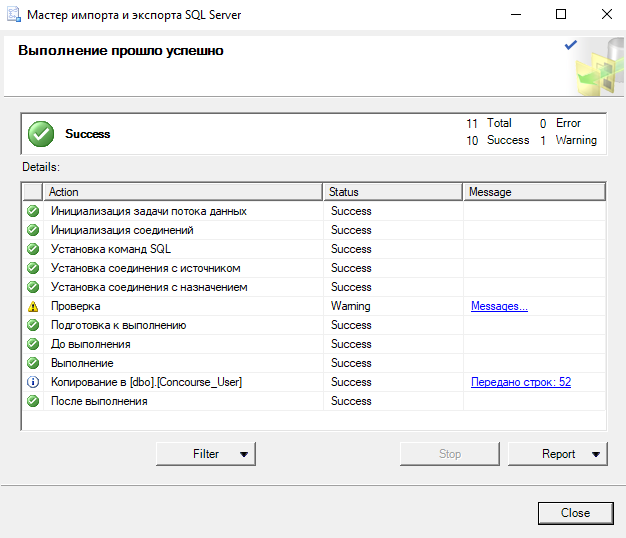


Рисунок 4 – Импорт, шаг 2

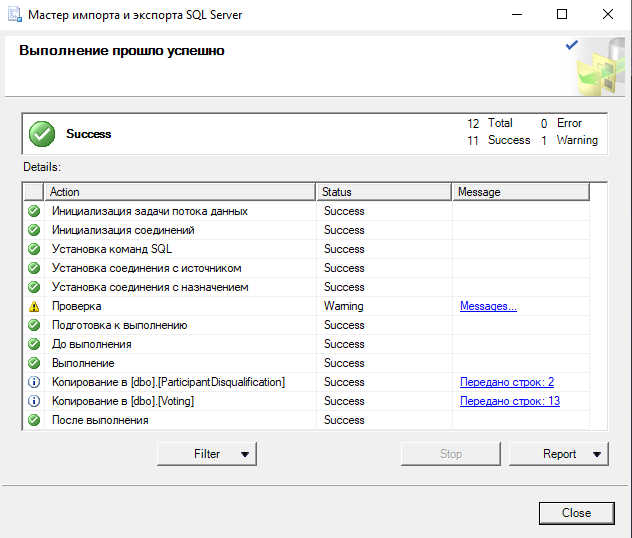


Рисунок 5– Импорт, шаг 3

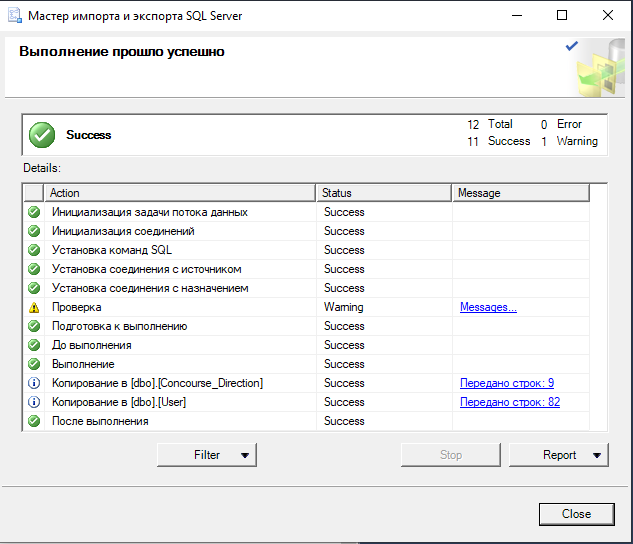


Рисунок 6 – Импорт, шаг 4

# 4. Разработка приложения

## 4.1 Разработка стиля

Каждое окно состоит из трех частей – верхней панели, боковой панели и рабочей области.

Верхняя панель включает в себя логотип и заголовок формы. Боковая панель содержит кнопки навигации. Рабочая область содержит фрейм для смены страниц.

Логотип, представленный на рисунке 7, можно описать как синий круг с оранжевой обводкой, внутри которого находится микрофон и символы нот.

Рисунок 7 - логотип

В качестве цветовой гаммы приложения были выбраны четыре цвета – синий, желтый, черный и белый.

Синий цвет используется в качестве фона для верхней и боковой панелей форм приложения, а также в качестве цвета для шрифта некоторых текстовых блоков. Желтый используется в качестве фона для кнопок. Черный цвет используется в качестве цвета шрифта текстовых полей, а также для вывода данных в таблицах. Белый цвет выступает в качестве фона для фона рабочей области окон и страниц.

Для шрифтов выбраны два семейства шрифта: Roboto Mono для текстовых блоков и данных в таблицах и Arial Black для заголовков.

Размер шрифта установлен следующим образом: 56 – для заголовков форм, 32 – для кнопок, 24 – для текстовых блоков и полей, 16 – для строк в таблицах.

Кнопки имеют 2 образца для размера – кнопка действия и кнопка навигации. Кнопки действия имеют размер 150x70 пикселей, а кнопки навигации 250x100 пикселей.

## 4.2 Разработка главного окна системы

Главное окно системы (MainWindow), представленное на рисунке x, разработано в соответствии со стилем приложения. Для создания окна, необходимо нажать ПКМ по проекту, выбрать "Добавить", "Окно" (см. рисунок 8). В всплывающем окне снизу указать название окна, нажать "Добавить" (см. рисунок 9). В файл MainWindow.xaml, расположенный в обозревателе решений, необходимо скопировать xaml код из приложения Б, а в MainWindow.xaml.cs C# код.

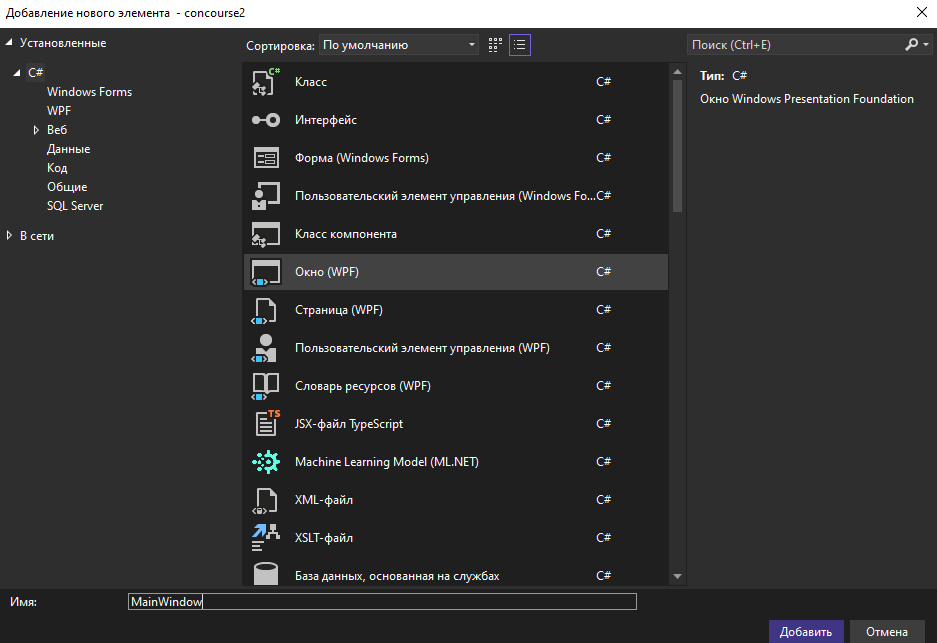
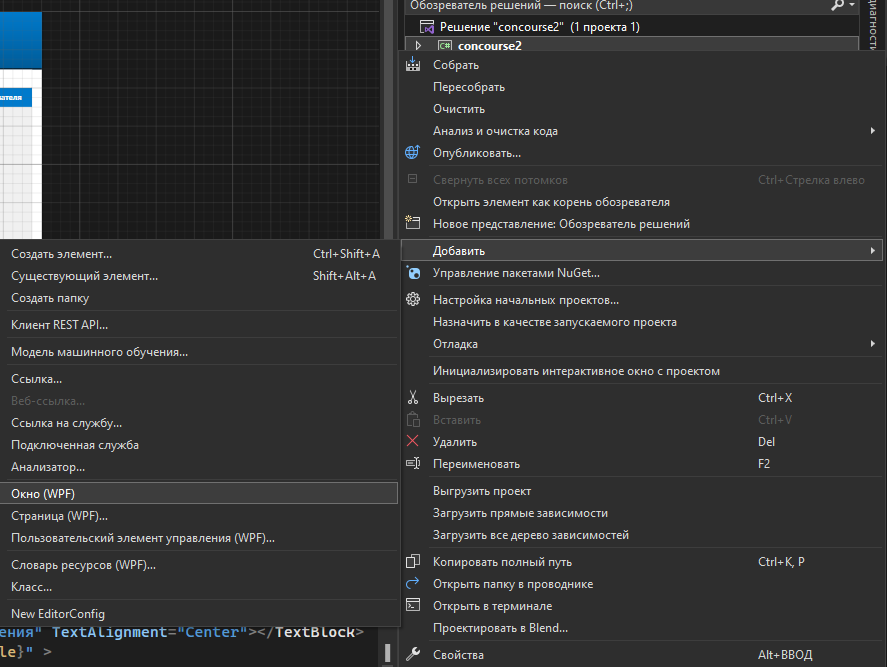
Рисунок 8 – Добавление нового окна в проект

Рисунок 9 – Указание название окна

Страница регистрации (RegistrationPage)

Создание страницы аналогично созданию окна. Код страницы регистрации представлен в Приложении Г.

Страница просмотра конкурсов (PerfomancePage)

Создание страницы аналогично созданию окна. Код страницы просмотра конкурсов представлен в Приложении В.

## 4.3 Разработка системы авторизации

Авторизация представлена отдельным окном (AuthWindow), переход к которому осуществляется по нажатию на кнопку "Авторизация", расположенную в главном окне. Размер окна 500x600 пикселей. В окне расположено две кнопки "Войти" и "Главная форма". По нажатию на кнопку "Войти" проверяется наличие в базе данных пользователя с введенным логином и паролем. Если пользователь не найден, то выводится сообщение об ошибке, если пользователь найден, то всплывающее окно Авторизации закрывается и данные об авторизации сохраняются в классе Auth. При нажатии на кнопку "Главная форма" всплывающее окно закрывается и пользователь попадает на "Главное окно".

Код класса авторизации:

public class Auth

{

public string UserType { get; set; }

public int UserId { get; set; }

}

Создание окна аналогично созданию окна MainWindow.

## 4.4 Разработка окон для работы пользователей

Формы пользователей (Организатор – OrganizerWindow, руководитель – ModeratorWindow, жюри – JuryWindow, участник – ParticipantWindow) разработаны в соответствии со стилем приложения.

При открытии окна пользователя в рабочей области отображается страница с персональными данными пользователя (PersonalCabinetPage).

Пользователь может изменять свои персональные данные и сохранить изменения по нажатию на кнопку "Сохранить". Также пользователь может изменить свой аватар.

Окно организатора

Создание окна аналогично созданию окна MainWindow.

Страница работы с конкурсами (ConcourseWork)

Создание страницы работы с конкурсами аналогично созданию окна.

Страницы работы с жюри и модераторами (UserWork)

Создание страницы работы с жюри и модераторами аналогично созданию окна.

Страница работа с направлениями (DirectionWork)

Создание страницы работы с направлениями аналогично созданию окна.

Страница работы с направлениями конкурса (ConcourseDirectionWork)

Создание страницы работы с направлениями конкурса аналогично созданию окна.

Страница работы с пользователями направления (ConcourseUserPage)

Создание страницы работы с пользователями направления аналогично созданию окна.

Окно Модератора

Создание окна аналогично созданию окна MainWindow.

Страница участники модератора (ModeratorConcourseUsersPage)

Создание страницы участники модератора аналогично созданию окна.

Окно Жюри.

Создание окна аналогично созданию окна MainWindow.

Страница Участники жюри (JuryDirectionPage)

Аналогична странице модератора

Окно участника

Создание окна аналогично созданию окна MainWindow.

Страница Направления участника (ParticipantConcourseInfo)

Аналогична странице модератора.

Окно прикрепления

Создание окна аналогично созданию окна MainWindow.

Окно дисквалификации

Создание окна аналогично созданию окна MainWindow.

# 5. Разработка руководства пользователя

1. Данное руководство предназначено для следующих пользователей :

* гость – неавторизованный пользователь, желающий просмотреть информацию по проходящему и прошедшим конкурса;
* организатор – пользователь, который осуществляет работу с данными конкурсов и направлений, регистрацию жюри и модераторов, а также их прикрепление;
* модератор – пользователь, ответственный за окончание проведения направлений конкурсов, установке отметок о выступлении участников и их дисквалификацию по своему направлению;
* жюри – пользователь, который осуществляет голосование за участников по своему направлению;
* участник – пользователь, который может просмотреть данные о других участниках и результатах своего выступления в рамках своего жанра.

1. Персональные данные

Для того, чтобы открыть страницу персональных данных(см. рисунок 10) пользователь должен авторизоваться под его логином и паролем (см. пункт 3), после чего нажать на кнопку "Личный кабинет", расположенную в главном окне (см. рисунок 12).

Для того чтобы изменить персональные данные необходимо ввести новые значения в текстовые поля, после чего нажать кнопку "Сохранить".

Для изменения аватара необходимо дважды нажать по изображению на странице персональных данных (Или же по квадрату, в центре который надпись "Выберите изображение"), после чего выберите необходимое изображение в проводнике (см. рисунок 11) и нажмите кнопку "Открыть".

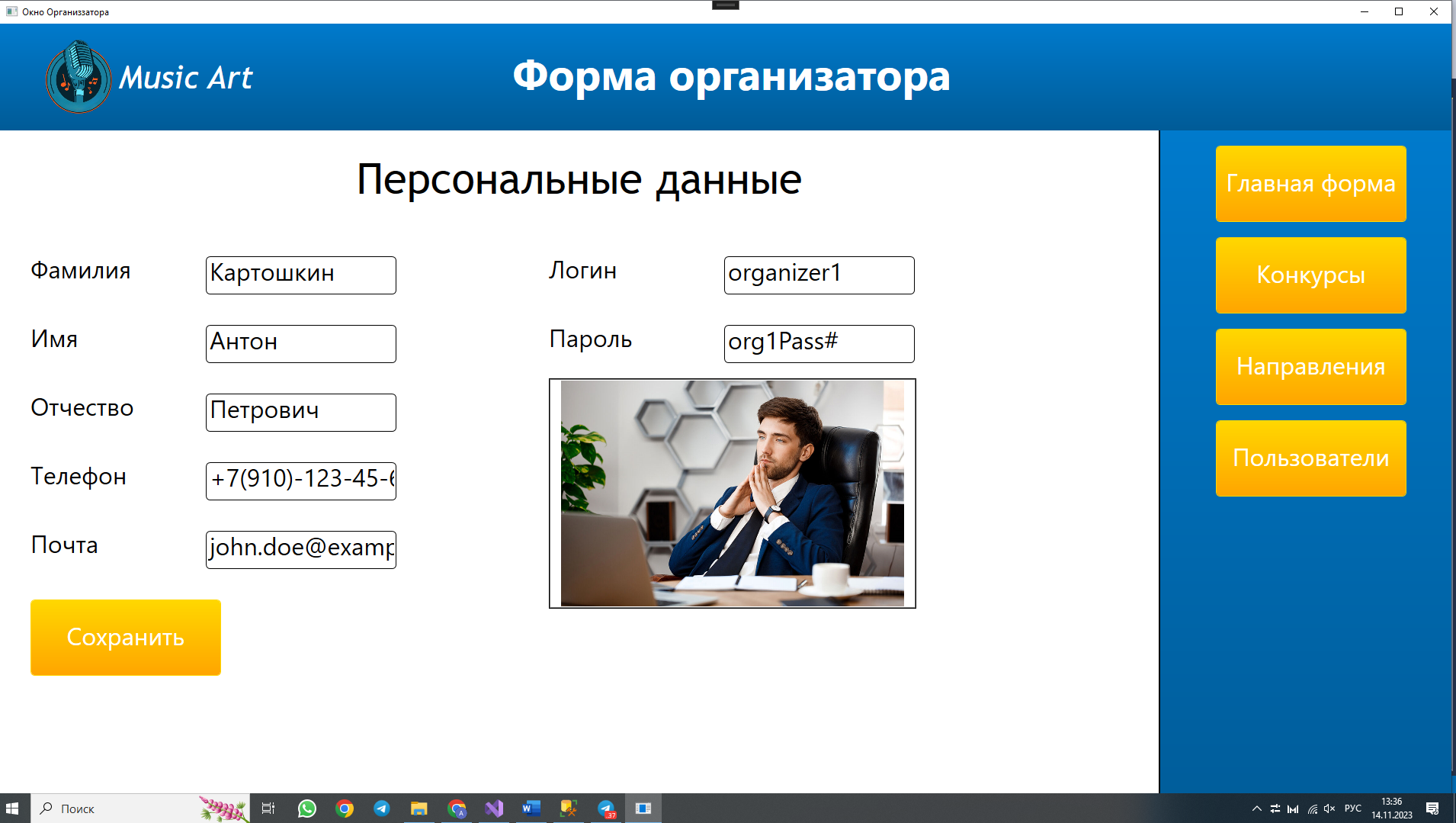


Рисунок 10 – Персональные данные

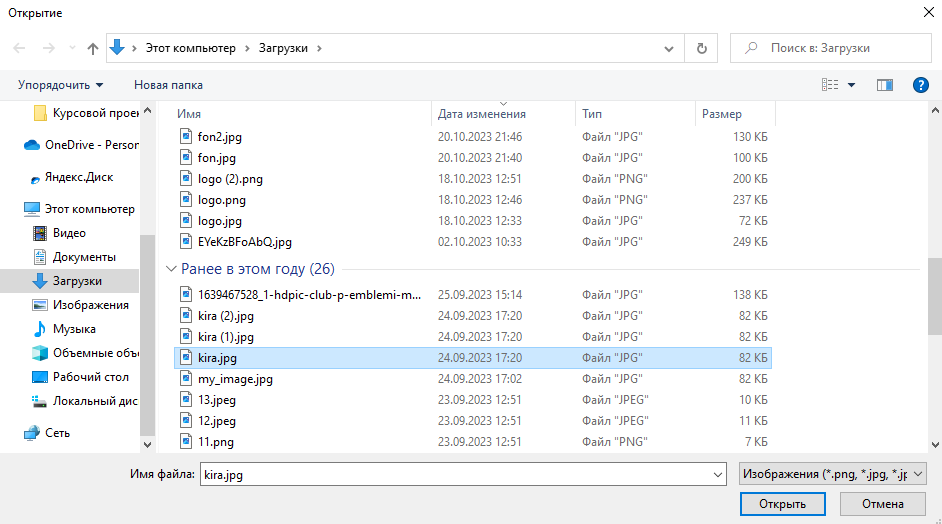
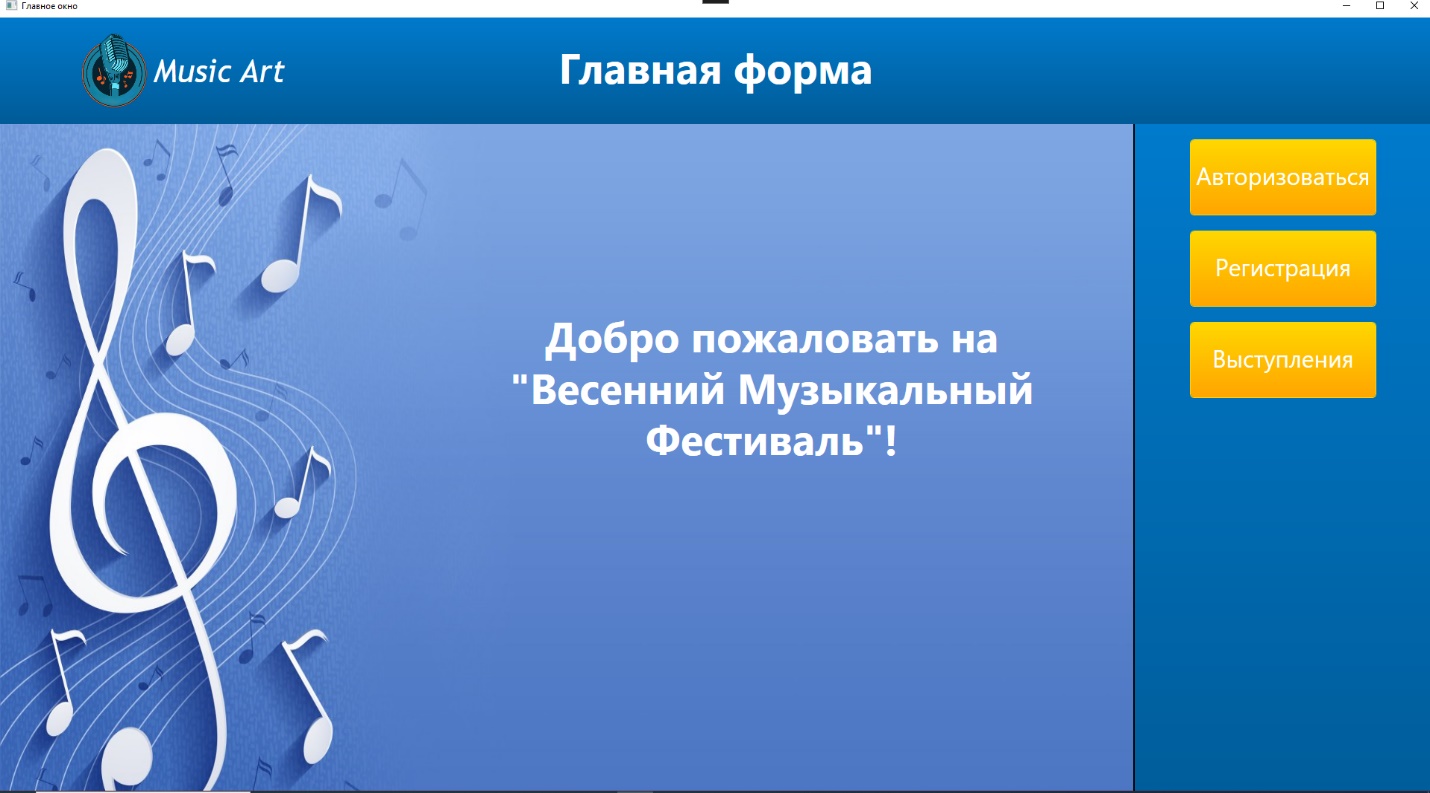


Рисунок 11 – Выбор аватара

1. Авторизация и выход из аккаунта

Для того чтобы авторизоваться под своей учетной записью необходимо в главном окне системе (см. рисунок 12) нажать по кнопке "Авторизация". В всплывающем окне (см. рисунок 13) указать свой логин и пароль в соответствующих текстовых полях и нажать кнопку "Войти".

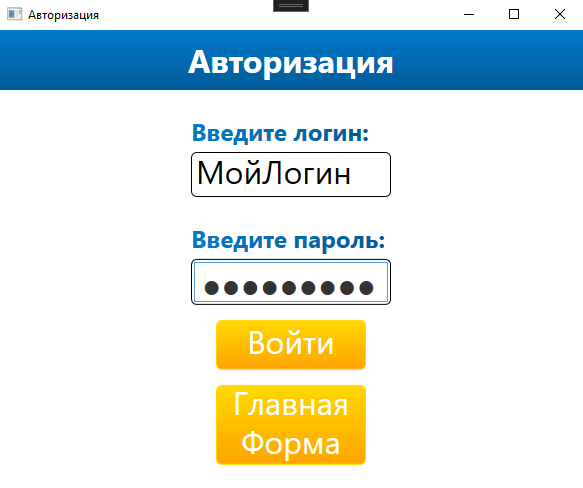
Рисунок 12 – Главное окно системы

Рисунок 13 – Окно авторизации

Для выхода из аккаунта необходимо нажать кнопку "Выйти".

1. Навигация

Для возврата на предыдущую страницу необходимо нажать кнопку "Назад".

Для перехода в главное окно системы из других окон необходимо нажать кнопку "Главная форма".

1. Организатор

Для начала работы необходимо авторизоваться в главном окне системы (см. пункт 3)

Для изменения и просмотра персональных данных см. пункт 2.

Работа с конкурсами

В окне Организатора (см. рисунок 14) нажмите по кнопке "Конкурсы", после чего откроется страница для работы с конкурсами (см. рисунок 15)

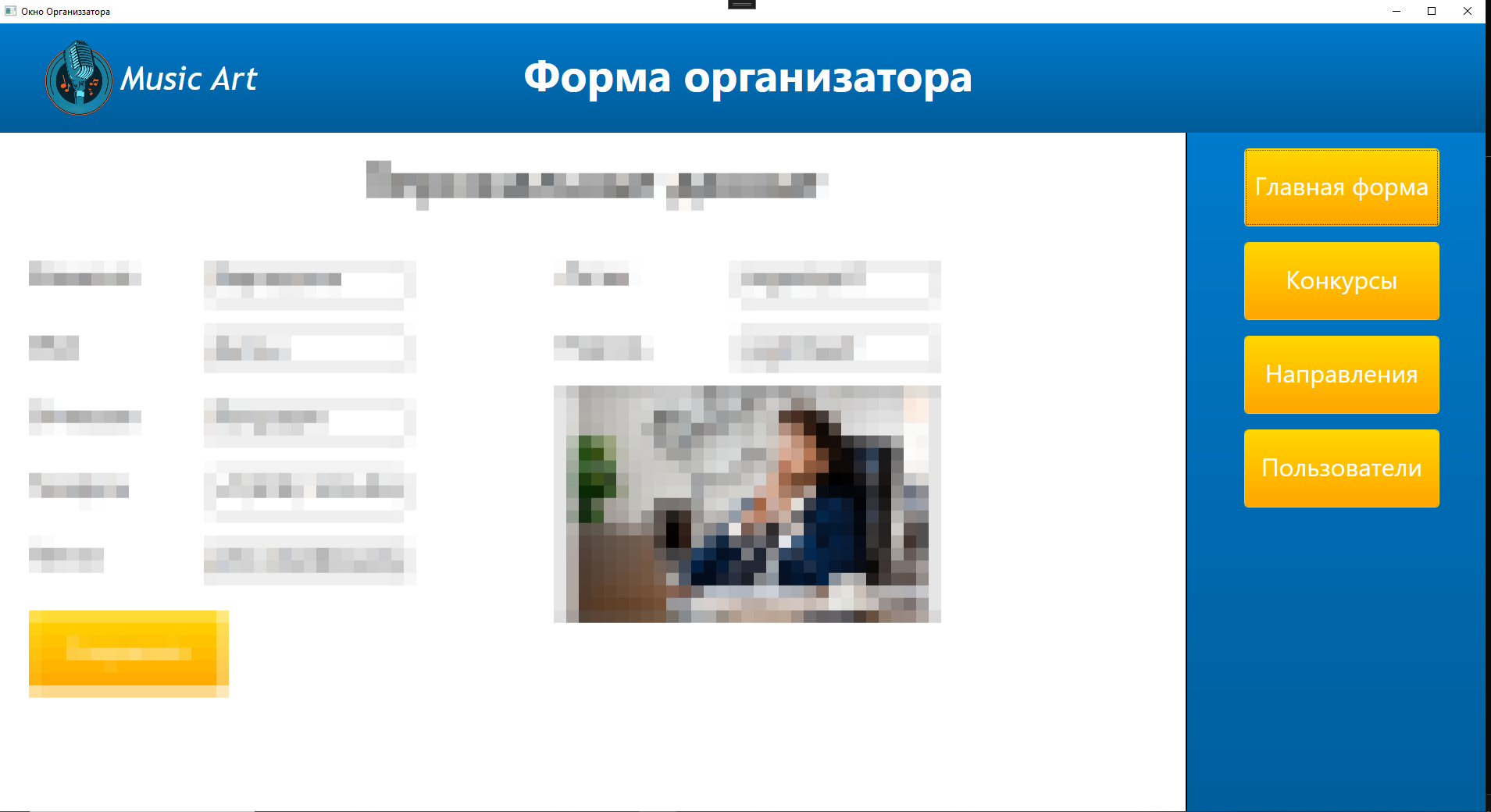


Рисунок 14 – Окно организатора

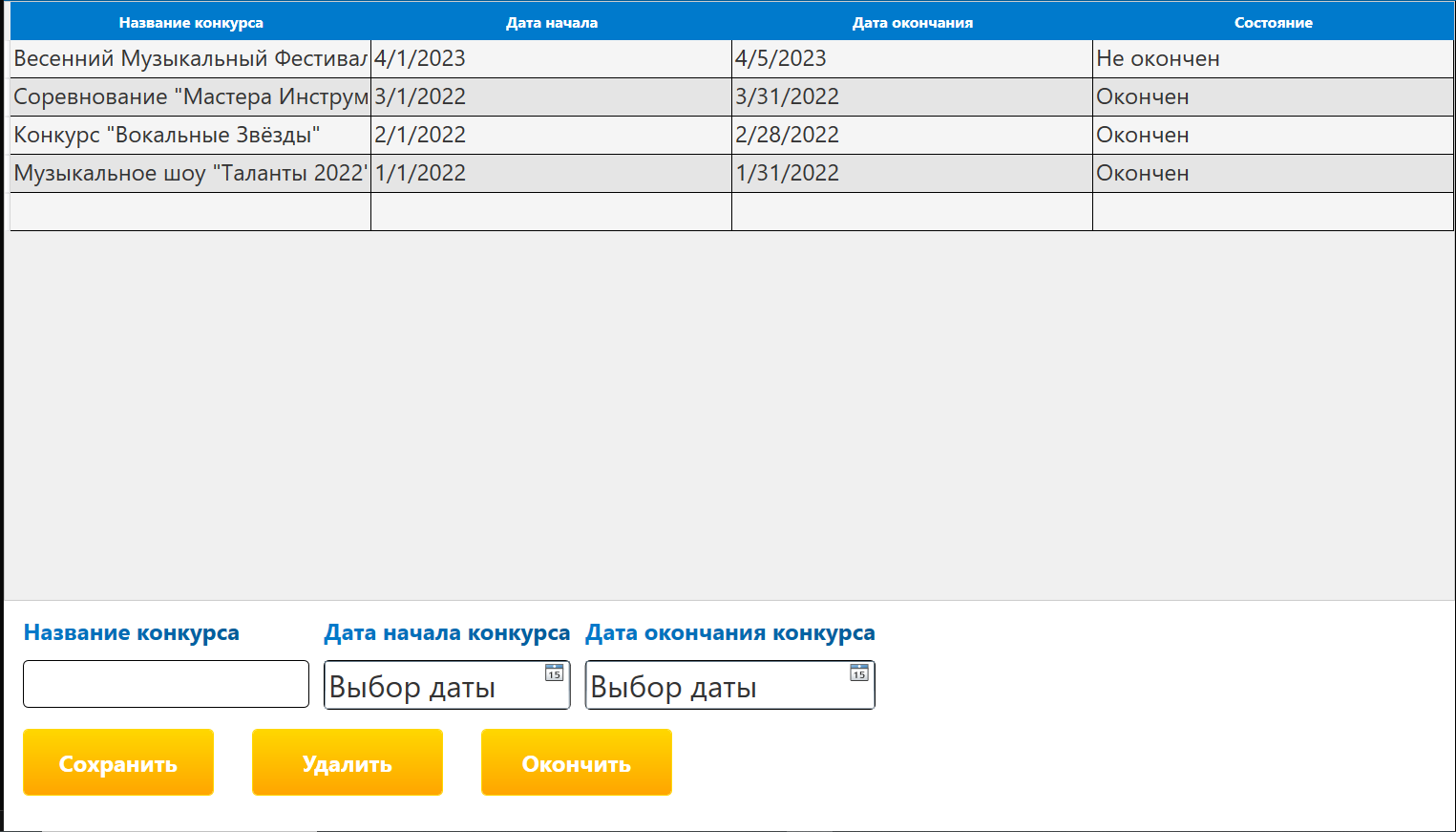


Рисунок 15 - Страница работы с конкурсами

Добавление конкурсов - Выберите нижнюю пустую строку в таблице, введите значения в текстовые поля, расположенные под таблицей: "Название конкурса", "Дата начала конкурса", "Дата окончания конкурса". Нажмите кнопку сохранить. Примечание: Дата окончания конкурса должна быть позже даты начала.

Изменение данных конкурса - выберите необходимый конкурс в таблице, введите новые значения в текстовые поля. Нажмите кнопку сохранить. Примечание: Дата окончания конкурса должна быть позже даты начала.

Удаление конкурса - Выберите необходимый конкурс в таблице. Нажмите кнопку "Удалить". Примечание: конкурс для удаления не должен иметь прикрепленных направлений.

Окончание конкурса - Выберите необходимый конкурс в таблице. Нажмите кнопку "Окончить".

Работа с направлениями

В окне Организатора (см. рисунок 14) нажмите по кнопке "Конкурсы", после чего откроется страница для работы с направлениями (см. рисунок 16)

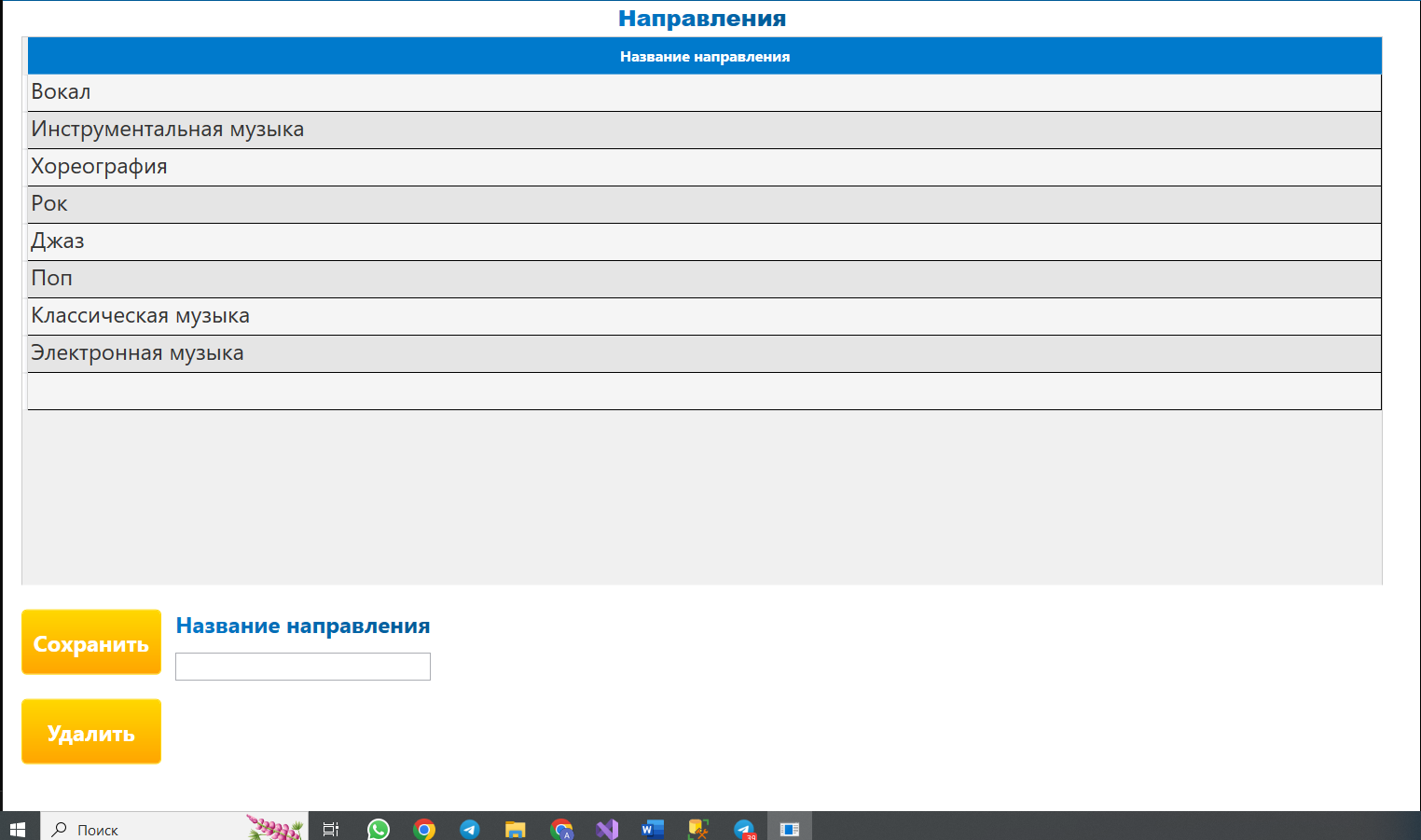


Рисунок 16 – Страница работы с направлениями

Добавление направления – Выберите нижнюю пустую строку в таблице, введите значение в текстовое поле, расположенные под таблицей - "Название направления". Нажмите кнопку сохранить.

Изменение данных – введите новое значение в текстовое поле. Нажмите кнопку сохранить.

Удаление направления – Выберите необходимое направление в таблице. Нажмите кнопку "Удалить". Примечание: направление для удаления не должно использоваться ни в одном конкурсе.

Работа с направлениями конкурса

На странице "Конкурсы" выберите конкурс, с направлениями которого хотите работать. Нажмите кнопку "Направления", после чего откроется страница "Направления конкурса, представленная на рисунке 17.

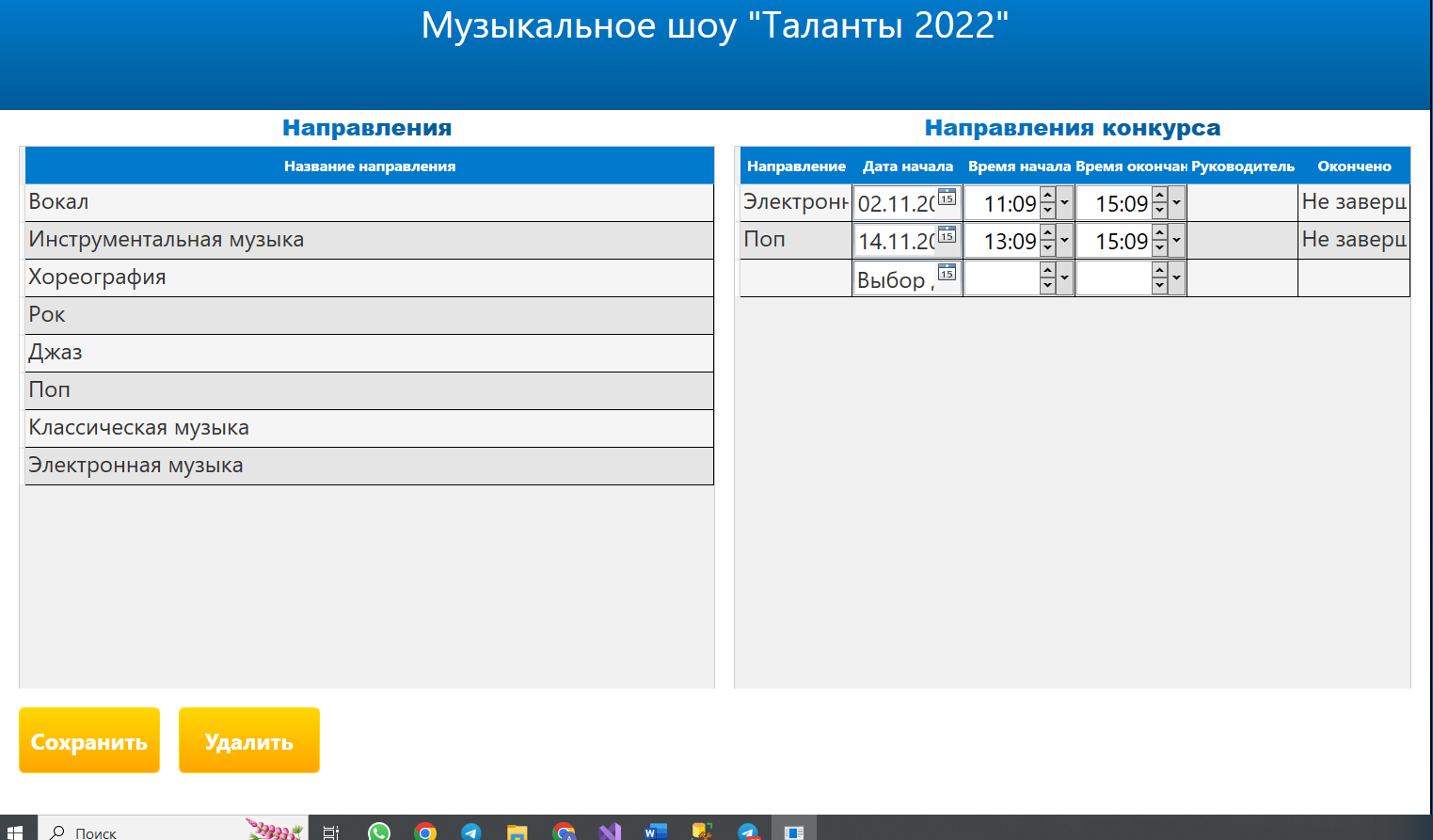


Рисунок 17 – Страница работы с направлениями конкурса

Добавление направления – сделайте двойной клик по направлению в таблице "Направления", которое хотите прикрепить. Выделите новое направление в таблице Направления конкурса. Нажмите кнопку сохранить.

Изменение данных конкурса – Выберите необходимое направление в таблице "Направления конкурса", введите новые значения в полях. Нажмите кнопку сохранить.

Удаление направления – Выберите необходимое направление в таблице. Нажмите кнопку "Удалить". Примечание: направление для удаления не должно содержать прикрепленных пользователей.

Прикрепление жюри и модераторов

На странице "Направления конкурса" выберите направление, к которому хотите прикрепить жюри или модераторов. Нажмите кнопку "Пользователи", после чего откроется страница "Пользователи конкурса", представленная на рисунке 18.

Прикрепление жюри или модератора – сделайте двойной щелчок по необходимому жюри или модератору в таблице "Пользователи". Нажмите кнопку "Сохранить". Примечание: к одному направлению можно прикрепить только одного модератора. Жюри и модератор может прикрепиться лишь к одному направлению в рамках конкурса.

Рисунок 18– Страница прикрепления жюри или модератора

Открепление жюри или модератора – выберите необходимого жюри или модератора в таблице "Пользователи направления", нажмите кнопку "Открепить".

1. Модератор

Для начала работы необходимо авторизоваться в главном окне системы (см. пункт 3)

Для изменения и просмотра персональных данных см. пункт 2.

Для начала осуществления своей работы необходимо перейти на страницу с участниками (см. рисунок 20), нажав по навигационной кнопке "Участники" из окна Модератора (см. рисунок 19).

Дисквалификация участника – выберите необходимого участника в таблице, нажмите кнопку "Дисквалифицировать". В всплывающем окне в текстовом поле укажите причину дисквалификации. Нажмите кнопку "Дисквалифицировать".

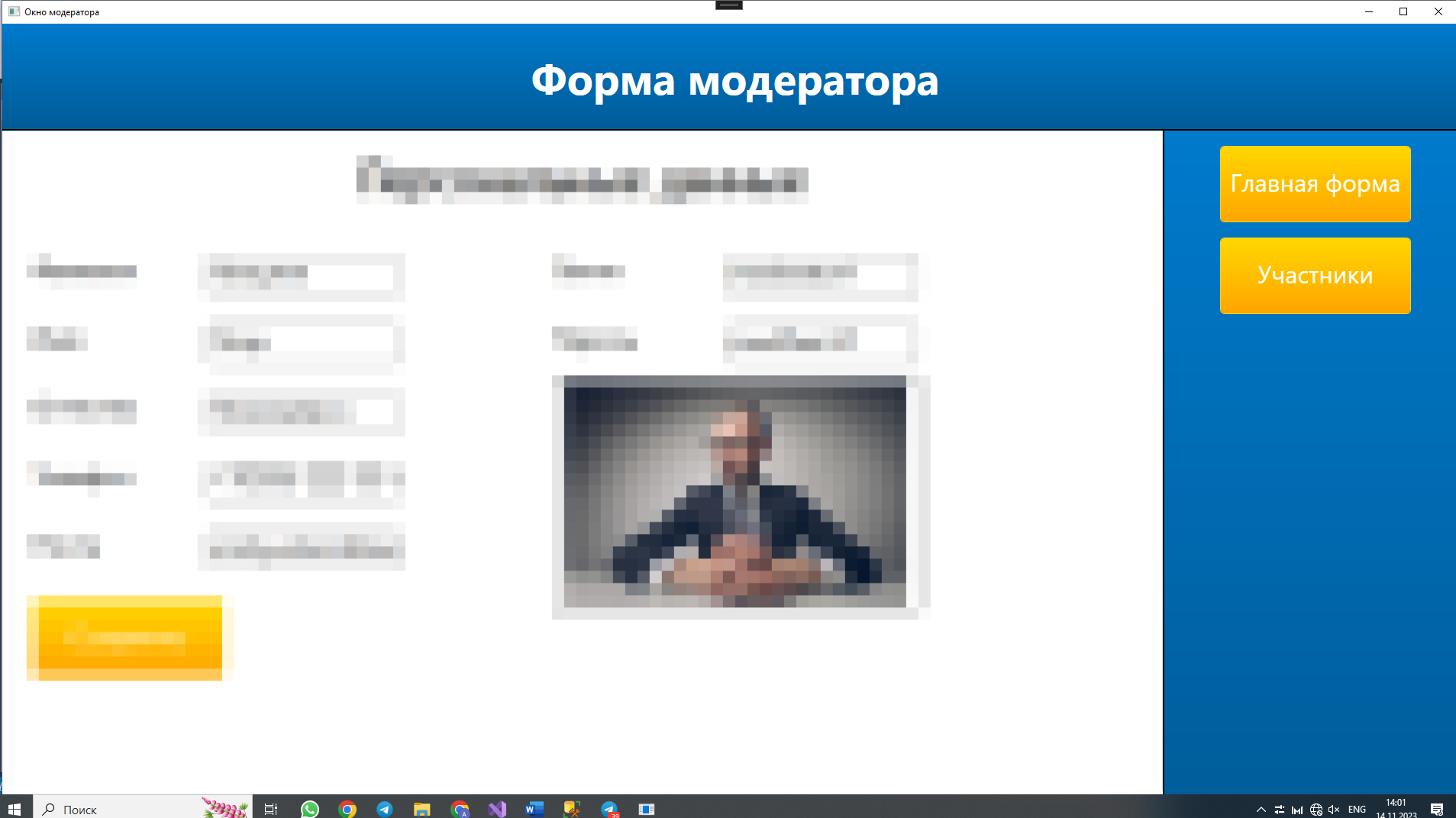


Рисунок 19 – Форма модератора

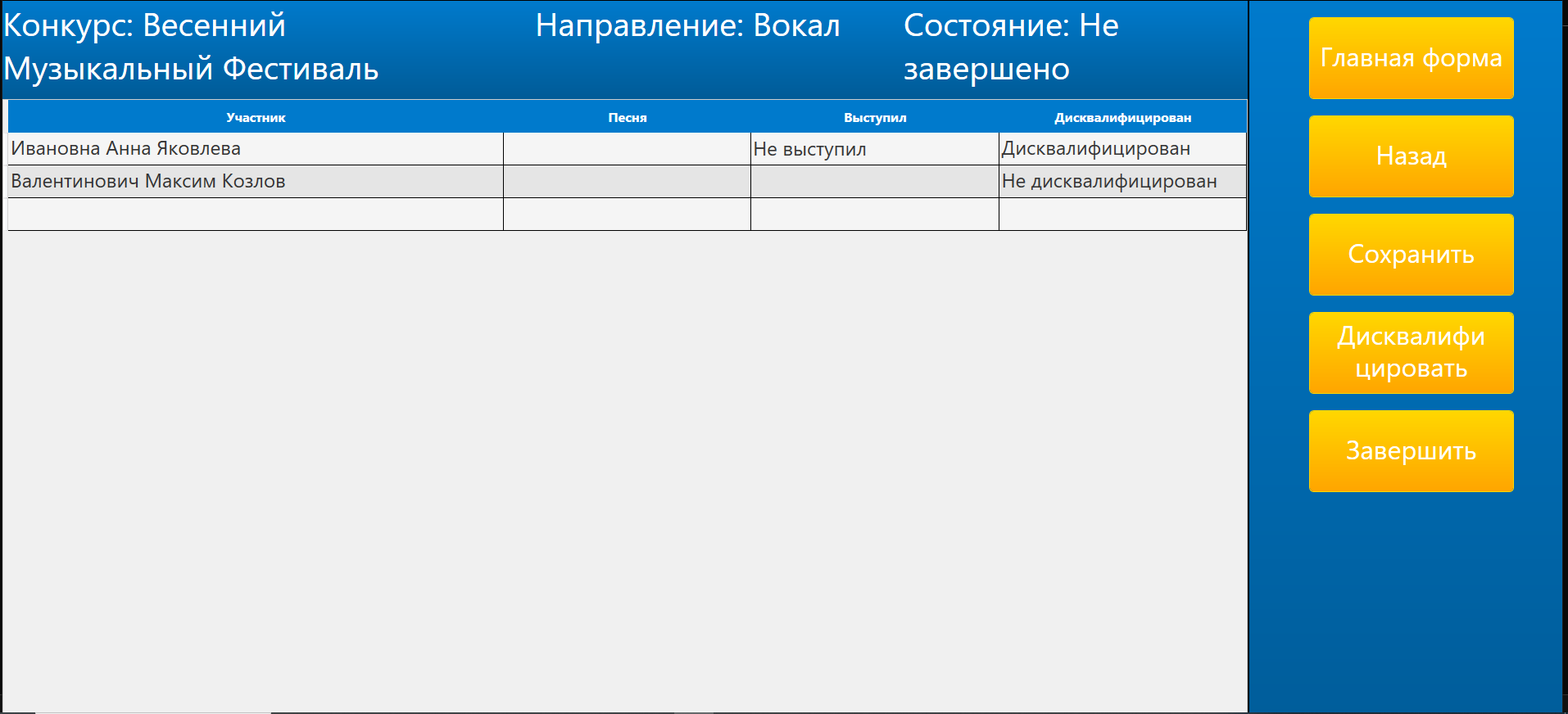


Рисунок 20 – Страница участники модератора

Отметка о выступлении участника – выберите необходимого участника в таблице, выберите поле "Выступил" в таблице, в выпадающем списке выберите "Выступил". Нажмите кнопку Сохранить.

Завершение направления конкурса – нажмите кнопку "Завершить".

1. Жюри

Для начала работы необходимо авторизоваться в главном окне системы (см. пункт 3)

Для изменения и просмотра персональных данных см. пункт 2.

Для начала осуществления своей работы необходимо перейти на страницу с участниками (см. рисунок 22), нажав по навигационной кнопке "Голосование" в форме Жюри (см. рисунок 21)

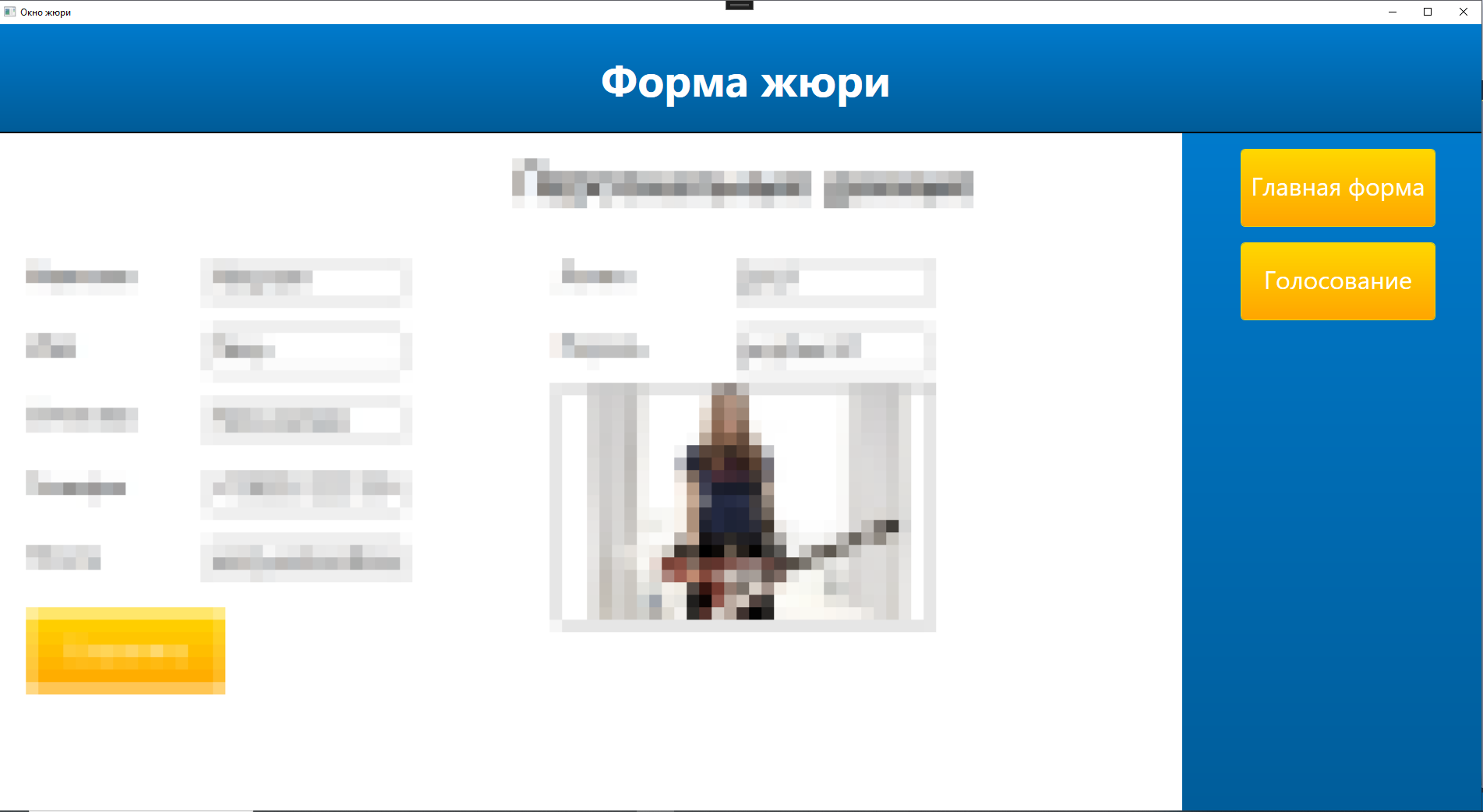


Рисунок 21 – Форма жюри

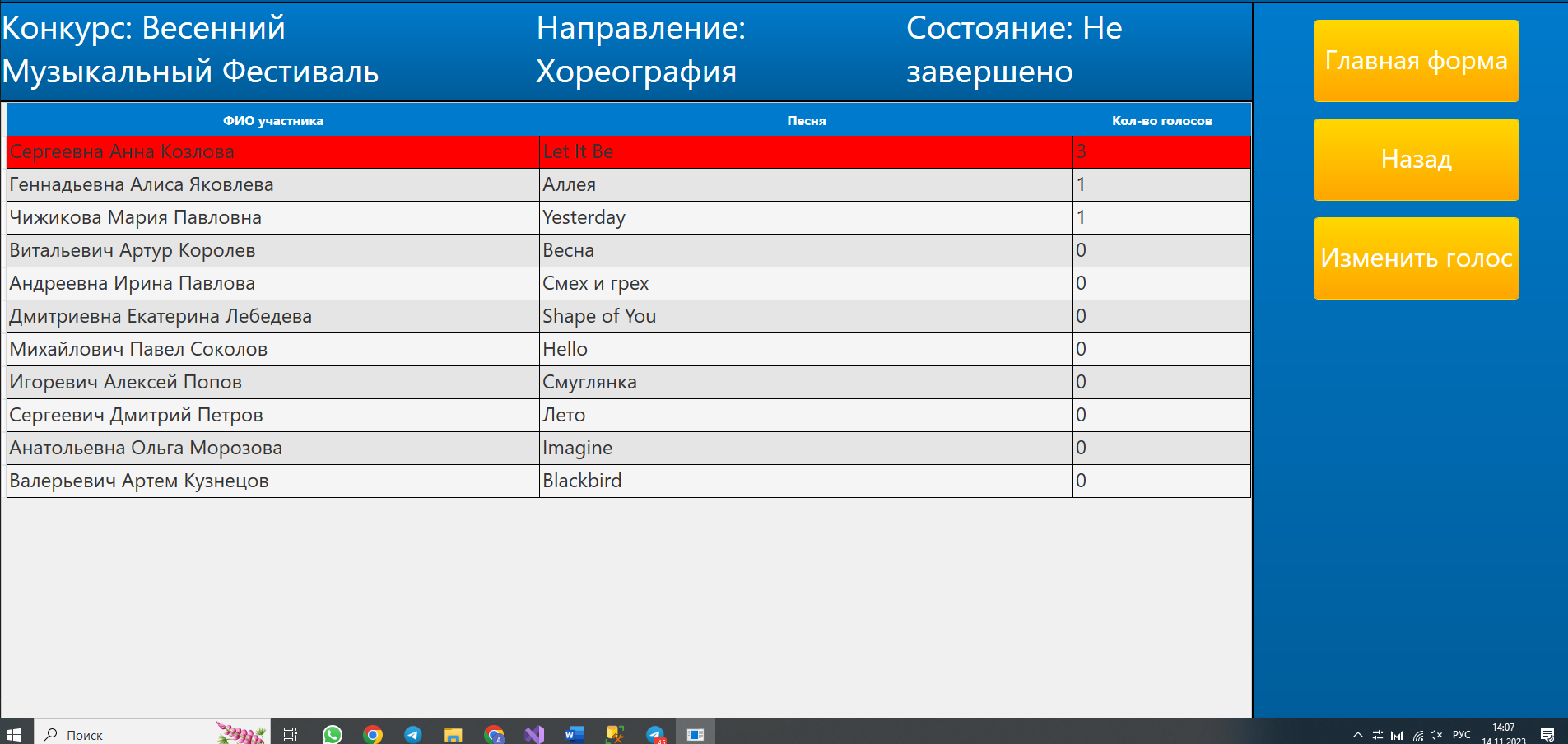


Рисунок 22 – Страница голосования

Голосование – выберите необходимого участника в таблице, нажмите кнопку "Проголосовать". В случае если ваш голос уже был отдан другому участнику по ошибке, вместо кнопки "Проголосовать" нажмите "Изменить голос".

Участник, за которого вы отдали голос, будет подсвечен красным в таблице.

1. Участник

Для начала работы необходимо авторизоваться в главном окне системы (см. пункт 3)

Для изменения и просмотра персональных данных см. пункт 2.

Для прикрепления в качестве участника к текущему конкурсу необходимо на форме Участника (см. рисунок 23) нажать по кнопке "Прикрепиться". В всплывающем окне (см. рисунок 24) выберите направление, к которому хотите прикрепиться из раскрывающегося списка, в поле ниже введите песню, с которой будете выступать. Нажмите кнопку "Прикрепиться".

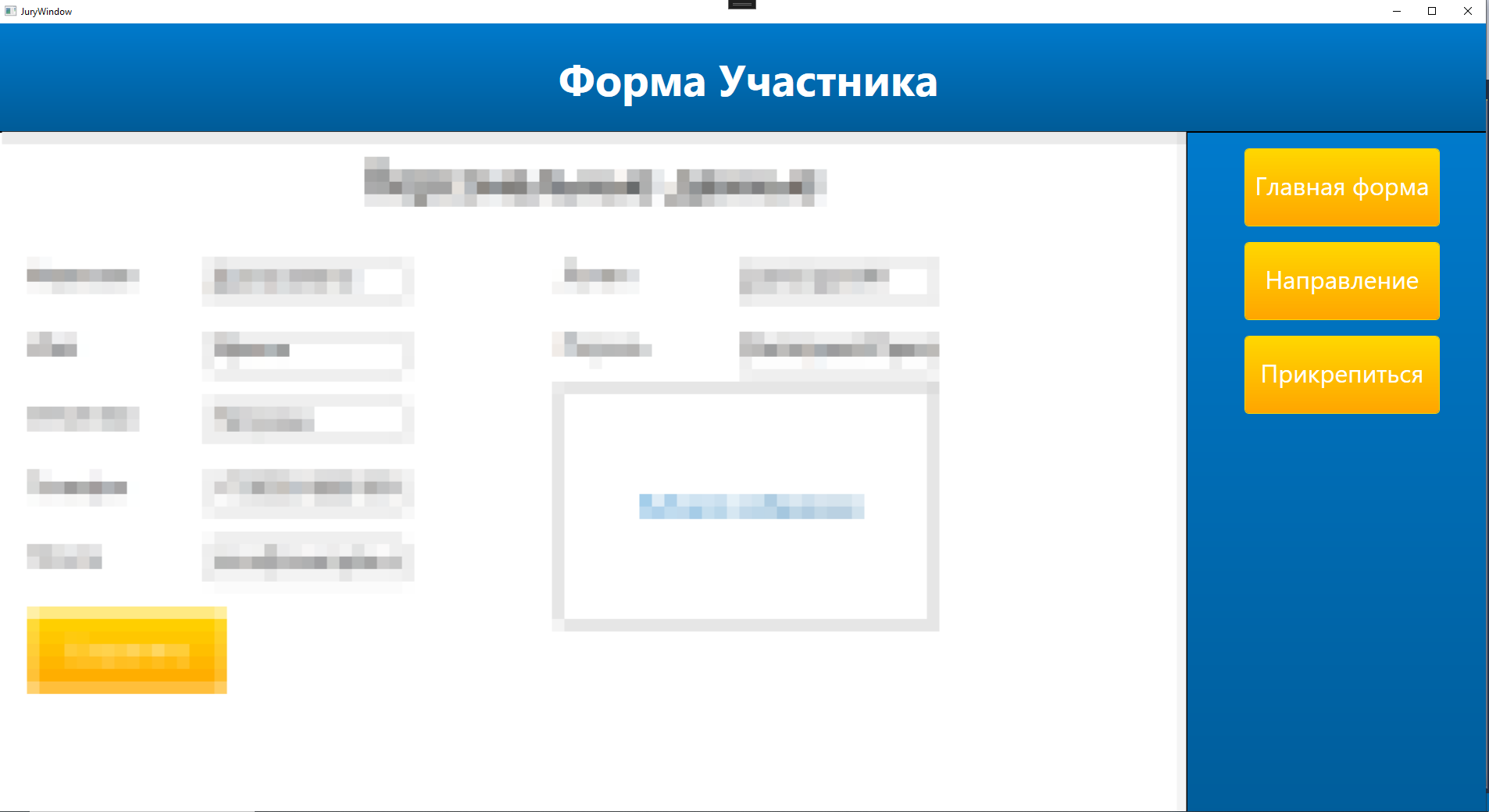


Рисунок 23 – Форма участника

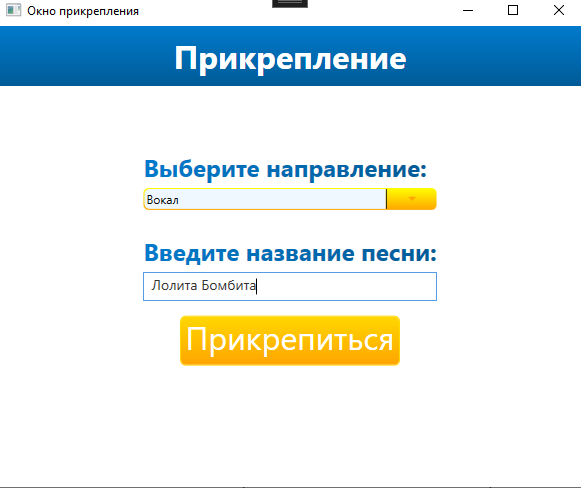


Рисунок 24– Окно прикрепления

Для просмотра сведений о направлении конкурса и его участниках, в котором вы в данный момент принимаете участие, нажмите по кнопке "Направление", после чего вы попадете на страницу "Направление участника", представленную на рисунке 25.



Рисунок 25– Страница направление участника

# Заключение

Результатом данного курсового проекта является автоматизированная информационная система, предназначенная для проведения музыкальных конкурсов на основе системы управления базами данных (СУБД). Во время работы был проведен анализ предметной области, проектирование и разработка базы данных, разработка приложения и руководства пользователя.

Разработанная система предоставляет удобное хранение данных о конкурсах и их просмотр, а также обеспечивает безопасную и быструю работу с конкурсами и направлениями для всех пользователей системы.

АИС имеет ряд преимуществ, среди которых:

1. Легко расширяемый код

2. Интуитивно понятный интерфейс

3. Безопасность данных

Также система имеет ряд перспектив развития:

1. Оптимизация SQL-запросов

2. Улучшение системы валидации данных

3. Внедрение системы ведения документации.

Таким образом данная работа действительно обладает существенной ценностью, программный продукт может быть интегрирован в деятельность организации и использоваться для решения перечисленных ранее задач.

# Информационные источники

1. MozhnoEge: [Электронный ресурс] Информационные системы и базы данных. URL: https://mozhnoege.ru/informatika/informatsionnye-sistemy-i-bazy-dannyh (Дата обращения: 29.10.2023)
2. GeekBrains: [Электронный ресурс] Основы работы с базами данных. URL: https://geekbrains.ru/courses/68 (Дата обращения: 30.10.2023)
3. Microsoft: [Электронный ресурс] Официальная документация Microsoft SQL Server. URL: https://docs.microsoft.com/en-us/sql/ (Дата обращения: 31.10.2023)
4. Coursera: [Электронный ресурс] Курс "Базы данных и SQL" на Coursera. URL: https://www.coursera.org/learn/sql-for-web-development (Дата обращения: 01.11.2023)
5. W3Schools: [Электронный ресурс] Руководство по SQL. URL: https://www.w3schools.com/sql/ (Дата обращения: 02.11.2023)
6. Docs.Microsoft: [Электронный ресурс] Документация по Windows Presentation Foundation (WPF). URL: https://docs.microsoft.com/en-us/dotnet/desktop/wpf/?view=netdesktop-7.0 (Дата обращения: 15.11.2023)
7. Троелсен, А. C# 8.0 и платформа .NET Core 3.0 / Санкт-Петербург: Питер, 2020. - 896 с.
8. Фримен, Э., WPF. Полное руководство / Москва: Вильямс, 2021. - 896 с.
9. Шилдт, Г. C# 9.0. Полное руководство / Санкт-Петербург: Питер, 2022. - 800 с.
10. Петцольд, Ч. WPF в действии / Москва: ДМК Пресс, 2020. - 650 с.

Приложение A

Скрипт создания базы данных:

CREATE DATABASE concourse;

GO

Use concourse

GO

CREATE TABLE [UserType](

[user\_type\_id] [int] PRIMARY KEY IDENTITY(1,1) NOT NULL,

[name] [nvarchar](30) NOT NULL,

);

CREATE TABLE [User](

[user\_id] [int] PRIMARY KEY IDENTITY(1,1) NOT NULL,

[f\_name] [nvarchar](30) NOT NULL,

[l\_name] [nvarchar](30) NOT NULL,

[patronymic] [nvarchar](30) NOT NULL,

[photo\_name] [nvarchar](50) NULL,

[phone] [nvarchar](20) NOT NULL,

[email] [nvarchar](50) NOT NULL,

[login] [nvarchar](50) NOT NULL UNIQUE,

[password] [nvarchar](50) NOT NULL,

[user\_type\_id] [int] FOREIGN KEY REFERENCES UserType(user\_type\_id) NOT NULL,

CONSTRAINT [CheckPassword] CHECK ((len([password])>=(6) AND [password] like '%[a-z]%' AND [password] like '%[A-Z]%' AND [password] like '%[0-9]%' AND [password] like '%[!@#%^&\*()]%')),

CONSTRAINT [CheckPhone] CHECK (([phone] like '+7([0-9][0-9][0-9])-[0-9][0-9][0-9]-[0-9][0-9]-[0-9][0-9]'))

);

CREATE TABLE [dbo].[Concourse](

[concourse\_id] [int] PRIMARY KEY IDENTITY(1,1) NOT NULL,

[name] [nvarchar](50) NOT NULL,

[start\_date] [date] NOT NULL,

[end\_date] [date] NOT NULL,

[is\_ended] [bit] DEFAULT ((0)) NOT NULL,

CONSTRAINT [CK\_Concourse] CHECK (([start\_date]<[end\_date])),

);

CREATE TABLE [Direction](

[direction\_id] [int] PRIMARY KEY IDENTITY(1,1) NOT NULL,

[name] [nvarchar](30) NOT NULL,

);

CREATE TABLE [dbo].[Concourse\_User](

[concourse\_user\_id] [int] PRIMARY KEY IDENTITY(1,1) NOT NULL,

[user\_id] [int] FOREIGN KEY REFERENCES [User]([user\_id]) NOT NULL,

[concourse\_direction\_id] [int] NULL,

[song] [nvarchar](50) NULL,

[isPerformed] [bit] NULL,

CONSTRAINT [IX\_Concourse\_Participants] UNIQUE NONCLUSTERED

(

[user\_id] ASC,

[concourse\_direction\_id] ASC

)

);

CREATE TABLE [dbo].[Concourse\_Direction](

[concourse\_direction\_id] [int] PRIMARY KEY IDENTITY(1,1) NOT NULL,

[concourse\_id] [int] FOREIGN KEY REFERENCES Concourse(concourse\_id) NOT NULL,

[direction\_id] [int] FOREIGN KEY REFERENCES Direction(direction\_id) NOT NULL,

[moderator\_id] [int] NULL,

[start\_date] [date] NOT NULL,

[start\_time] [datetime] NOT NULL,

[end\_time] [datetime] NOT NULL,

[is\_ended] [bit] DEFAULT(0) NOT NULL,

CONSTRAINT [IX\_Concourse\_Direction] UNIQUE NONCLUSTERED

(

[concourse\_id] ASC,

[direction\_id] ASC,

[moderator\_id] ASC

)

);

ALTER TABLE Concourse\_user

ADD CONSTRAINT FK\_ConcourseUser\_ConcourseDirection

FOREIGN KEY (concourse\_direction\_id)

REFERENCES [Concourse\_direction](concourse\_direction\_id);

ALTER TABLE Concourse\_direction

ADD CONSTRAINT FK\_ConcourseDirection\_ConcourseUser

FOREIGN KEY (moderator\_id)

REFERENCES [Concourse\_user](concourse\_user\_id);

ALTER TABLE Concourse\_direction

ADD CONSTRAINT FK\_ConcourseDirectionWinner\_ConcourseUser

FOREIGN KEY (winner\_id)

REFERENCES [Concourse\_user](concourse\_user\_id);

CREATE TABLE [dbo].[ParticipantDisqualification](

[disqualification\_id] [int] PRIMARY KEY IDENTITY(1,1) NOT NULL,

[participant\_id] [int] FOREIGN KEY REFERENCES [Concourse\_user](concourse\_user\_id) NOT NULL,

[concourse\_direction\_id] [int] FOREIGN KEY REFERENCES Concourse\_direction(concourse\_direction\_id) NOT NULL,

[motive] [nvarchar](100) NOT NULL,

CONSTRAINT [IX\_ParticipantDisqualification] UNIQUE NONCLUSTERED

(

[concourse\_direction\_id] ASC,

[participant\_id] ASC

)

);

CREATE TABLE [dbo].[Voting](

[voting\_id] [int] PRIMARY KEY IDENTITY(1,1) NOT NULL,

[concourse\_direction\_id] [int] FOREIGN KEY REFERENCES [Concourse\_direction](concourse\_direction\_id) NOT NULL,

[concourse\_jury\_id] [int] FOREIGN KEY REFERENCES [Concourse\_user](concourse\_user\_id) NOT NULL,

[concourse\_participant\_id] [int] FOREIGN KEY REFERENCES [Concourse\_user](concourse\_user\_id) NOT NULL,

CONSTRAINT [IX\_Voting] UNIQUE NONCLUSTERED

(

[concourse\_participant\_id] ASC,

[concourse\_jury\_id] ASC,

[concourse\_direction\_id] ASC

)

);

# Приложение Б

Код главного окна

XAML:

<Window x:Class="concourse2.MainWindow"

Title="Главное окно" WindowState="Maximized"ContentRendered="Window\_ContentRendered">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="140"></RowDefinition>

<RowDefinition Height="940"></RowDefinition>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="760"></ColumnDefinition>

<ColumnDefinition Width="760"></ColumnDefinition>

<ColumnDefinition Width="\*"></ColumnDefinition>

</Grid.ColumnDefinitions>

<Grid Grid.Row="0" Grid.Column="0" Grid.ColumnSpan="3" Style="{StaticResource MusicPanelStyle}">

<Grid>

<WrapPanel Margin="100 0 0 0" x:Name="logo" VerticalAlignment="Center">

<Image Source="Resources/logo.png" HorizontalAlignment="Left" Height="100"></Image>

<TextBlock Text="Music Art" VerticalAlignment="Center" Style="{StaticResource CompanyLogoStyle}"></TextBlock>

</WrapPanel>

<TextBlock Style="{StaticResource FormTitle}" HorizontalAlignment="Center" VerticalAlignment="Center" Text="Главная форма"></TextBlock>

</Grid>

</Grid>

<Grid Grid.Row="1" Grid.Column="0" Grid.ColumnSpan="2">

<Frame x:Name="mainFrame" NavigationUIVisibility="Hidden"></Frame>

</Grid>

<Grid Grid.Column="2" Grid.Row="1" Style="{StaticResource MusicPanelStyle}">

<Border BorderBrush="Black" BorderThickness="2,0,0,0">

<StackPanel >

<Button Margin="0 20 0 0" x:Name="authButton" Style="{StaticResource CustomMedimButtonStyle}" Content="Авторизоваться" Click="authButton\_Click"></Button>

<Button Margin="0 20 0 0" x:Name="regestrationButton" Style="{StaticResource CustomMedimButtonStyle}" Content="Регистрация" Click="regestrationButton\_Click"></Button>

<Button Margin="0 20 0 0" x:Name="openPersonalAccountButton" Style="{StaticResource CustomMedimButtonStyle}" Content="Личный кабинет" Click="openPersonalAccountButton\_Click" Visibility="Collapsed"></Button>

<Button Margin="0 20 0 0" x:Name="acconutExitButton" Style="{StaticResource CustomMedimButtonStyle}" Content="Выйти" Click="acconutExit\_Click" Visibility="Collapsed"></Button>

<Button Margin="0 20 0 0" x:Name="participantButton" Style="{StaticResource CustomMedimButtonStyle}" Content="Выступления" Click="participantButton\_Click"></Button>

<Button Margin="0 20 0 0" x:Name="goBackButton" Style="{StaticResource CustomMedimButtonStyle}" Content="Назад" Visibility="Collapsed" Click="goBackButton\_Click"></Button>

</StackPanel>

</Border>

</Grid>

</Grid>

</Window>

C#:

public partial class MainWindow : Window

{

// Объект для доступа к глобальным данным приложения

public GlobalDataService GlobalData { get; set; }

// Объект для доступа к базе данных

public dbContext DbContext { get; set; }

// Текущее имя страницы

public MainPageNames CurrentPageName { get; set; } = MainPageNames.WelcomePage;

// Предыдущее имя страницы

public MainPageNames PreviousPageName { get; set; }

// Конструктор по умолчанию

public MainWindow()

{

// Инициализация объекта базы данных

DbContext = new dbContext();

// Создание объекта для глобальных данных, если он еще не создан

if (GlobalData == null) GlobalData = new GlobalDataService();

// Установка объекта базы данных в глобальные данные

GlobalData.DbContext = DbContext;

// Инициализация компонентов окна

InitializeComponent();

// Навигация на страницу приветствия при запуске приложения

mainFrame.Navigate(new WelcomePage(GlobalData));

}

// Конструктор, принимающий глобальные данные

public MainWindow(GlobalDataService globalData) : this()

{

GlobalData = globalData;

GlobalData.DbContext = new dbContext();

}

// Обработчик события нажатия кнопки аутентификации

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

{

// Создание и отображение окна аутентификации

AuthWindow authWindow = new AuthWindow(GlobalData);

bool? authResult = authWindow.ShowDialog();

// Если аутентификация прошла успешно, сохранение информации об аутентификации

if (authResult == true)

{

GlobalData.AuthInfo = authWindow.AuthInfo;

}

// Проверка состояния аутентификации

IsAuth();

}

// Обработчик события нажатия кнопки перехода на страницу участников

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

{

// Установка текущего имени страницы

CurrentPageName = MainPageNames.UsersPage;

// Навигация на страницу участников

mainFrame.Navigate(new UsersPage(DbContext));

// Управление видимостью элементов

VisibilityControl();

}

// Метод для управления видимостью элементов в зависимости от текущей страницы

private void VisibilityControl()

{

// Сохранение предыдущего имени страницы

PreviousPageName = CurrentPageName;

// Управление видимостью элементов в зависимости от текущей страницы

switch (CurrentPageName)

{

case MainPageNames.WelcomePage:

// ...

break;

case MainPageNames.RegisterPage:

// ...

break;

case MainPageNames.UsersPage:

// ...

break;

}

}

// Метод для проверки состояния аутентификации и управления видимостью элементов

private void IsAuth()

{

// Проверка наличия информации об аутентификации

if (GlobalData.AuthInfo != null)

{

// Управление видимостью элементов для авторизованного пользователя

}

else

{

// Управление видимостью элементов для неавторизованного пользователя

}

}

// Обработчик события нажатия кнопки открытия личного кабинета

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

{

// Метод для открытия личного кабинета в зависимости от роли пользователя

OpenUserPersonalCabinet();

}

// Метод для открытия личного кабинета в зависимости от роли пользователя

private void OpenUserPersonalCabinet()

{

// Получение данных о текущем конкурсе

int currentConcourseId = ConcourseDBActions.GetLastConcourse(DbContext).concourse\_id;

// В зависимости от роли пользователя открытие соответствующего окна

switch (GlobalData.AuthInfo.UserType)

{

case "Организатор":

OpenOrganizerPersonalCabinet();

break;

case "Жюри":

ConcourseDirectionDataForUser juryCurrentConcourseDirection = ConcourseDirectionDBActions.GetConcourseDirectionDataForUser(DbContext, currentConcourseId, GlobalData.AuthInfo.UserId);

OpenJuryPersoncalCabinet(juryCurrentConcourseDirection);

break;

case "Руководитель":

ConcourseDirectionDataForUser moderatorCurrentConcourseDirection = ConcourseDirectionDBActions.GetConcourseDirectionDataForUser(DbContext, currentConcourseId, GlobalData.AuthInfo.UserId);

OpenModeratorPersoncalCabinet(moderatorCurrentConcourseDirection);

break;

case "Участник":

ConcourseDirectionDataForUser concourseDirectionDataForParticipant = ConcourseDirectionDBActions.GetConcourseDirectionDataForUser(DbContext, currentConcourseId, GlobalData.AuthInfo.UserId);

OpenParticipantPersoncalCabinet(concourseDirectionDataForParticipant);

break;

}

}

// Метод для открытия окна организатора

private void OpenOrganizerPersonalCabinet()

{

OrganizerWindow newWindow = new OrganizerWindow(GlobalData);

newWindow.Show();

this.Close();

}

// Метод для открытия окна жюри

private void OpenJuryPersoncalCabinet(ConcourseDirectionDataForUser data)

{

JuryDataService juryData = setJuryData(data);

JuryWindow juryWindow = new JuryWindow(GlobalData, juryData);

juryWindow.Show();

this.Close();

}

// Метод для открытия окна руководителя

private void OpenModeratorPersoncalCabinet(ConcourseDirectionDataForUser data)

{

ModeratorDataService moderatorData = setModeratorData(data);

ModeratorWindow moderatorWindow = new ModeratorWindow(GlobalData, moderatorData);

moderatorWindow.Show();

this.Close();

}

// Метод для открытия окна участника

private void OpenParticipantPersoncalCabinet(ConcourseDirectionDataForUser data)

{

ParticipantDataService participantData = new ParticipantDataService();

participantData.ConcourseDirectionInfo = data;

ParticipantWindow participantWindow = new ParticipantWindow(GlobalData, participantData);

participantWindow.Show();

this.Close();

}

// Метод для установки данных для окна жюри

private JuryDataService setJuryData(ConcourseDirectionDataForUser concourseDirectionJuryData)

{

JuryDataService juryData = new JuryDataService();

juryData.ConcourseDirection = concourseDirectionJuryData;

return juryData;

}

// Метод для установки данных для окна руководителя

private ModeratorDataService setModeratorData(ConcourseDirectionDataForUser concourseDirectionModeratorData)

{

ModeratorDataService moderatorData = new ModeratorDataService();

moderatorData.ConcourseDirecitonData = concourseDirectionModeratorData;

return moderatorData;

}

// Обработчик события завершения рендеринга контента окна

private void Window\_ContentRendered(object sender, EventArgs e)

{

// Проверка состояния аутентификации после рендеринга контента

IsAuth();

}

// Обработчик события нажатия кнопки выхода из аккаунта

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

{

// Сброс информации об аутентификации и управление видимостью элементов

GlobalData.AuthInfo = null;

IsAuth();

}

// Обработчик события нажатия кнопки регистрации

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

{

// Открытие формы регистрации

OpenRegisterForm();

}

// Метод для открытия формы регистрации

private void OpenRegisterForm()

{

// Установка текущего имени страницы

CurrentPageName = MainPageNames.RegisterPage;

// Навигация на страницу регистрации

mainFrame.Navigate(new RegistrationPage(GlobalData));

// Управление видимостью элементов

VisibilityControl();

}

// Обработчик события нажатия кнопки возврата

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

{

// Выполнение действий при возврате на предыдущую страницу

GoBack();

}

// Метод для выполнения действий при возврате на предыдущую страницу

private void GoBack()

{

// В зависимости от текущей страницы выполнение соответствующих действий

switch (CurrentPageName)

{

case MainPageNames.RegisterPage:

CurrentPageName = MainPageNames.WelcomePage;

VisibilityControl();

mainFrame.Navigate(new WelcomePage(GlobalData));

break;

case MainPageNames.UsersPage:

CurrentPageName = MainPageNames.WelcomePage;

VisibilityControl();

mainFrame.Navigate(new WelcomePage(GlobalData));

break;

}}}

# Приложение В

Код страницы просмотра конкурсов

XAML:

<Page x:Class="concourse2.Pages.GuestPages.UsersPage"

d:DesignHeight="940" d:DesignWidth="1520"

Title="PerfomancePage">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="100"></RowDefinition>

<RowDefinition Height="\*"></RowDefinition>

</Grid.RowDefinitions>

<StackPanel Grid.Row="0" Height="120" Orientation="Horizontal" Style="{StaticResource MusicPanelStyle}" HorizontalAlignment="Stretch">

<TextBlock Margin="45 0 0 0" TextWrapping="Wrap" x:Name="concourseNameTextBlock" FontSize="40" Foreground="white" Text="Конкурс:"></TextBlock>

</StackPanel>

<Grid Grid.Row="1" Background="White">

<Grid.RowDefinitions>

<RowDefinition Height="160"></RowDefinition>

<RowDefinition Height="\*"></RowDefinition>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition></ColumnDefinition>

</Grid.ColumnDefinitions>

<Grid Grid.Row="0">

<StackPanel>

<TextBlock TextAlignment="Center" Style="{StaticResource MyTextBlockStyle}" Text="Фильтры"></TextBlock>

<WrapPanel Margin="0 10 0 0">

<ComboBox Margin="50 0 0 0" x:Name="directionComboBox" Width="300" Height="100" Style="{StaticResource ComboBoxTest2}" DisplayMemberPath="Direction.name" SelectedValuePath="concourse\_direction\_id" SelectionChanged="directionComboBox\_SelectionChanged"></ComboBox>

<StackPanel Margin="40 0 0 0"/>

<CheckBox x:Name="onlyWinners" Style="{StaticResource CustomCheckBoxStyle}" FontSize="24" Margin="40 0 0 0" Height="30" Content="Только победители" Click="onlyWinners\_Click"></CheckBox>

<Button Margin="525 0 0 0" x:Name="clearFilters" Style="{StaticResource CustomMedimButtonStyle}" Content="Очистить" HorizontalAlignment="Left" Click="clearFilters\_Click"></Button>

</WrapPanel>

</StackPanel>

</Grid>

<Grid Grid.Row="1">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="50"></ColumnDefinition>

<ColumnDefinition Width="1400"></ColumnDefinition>

<ColumnDefinition Width="50"></ColumnDefinition>

</Grid.ColumnDefinitions>

<Button x:Name="goLeft" Content="&lt;" Grid.Column="0" Height="70" FontSize="40" Click="changeConcourse\_Click"></Button>

<DataGrid AutoGenerateColumns="False" Style="{StaticResource CustomDataGridStyle}" VerticalAlignment="Top" x:Name="participantsListView" Height="550" Grid.Column="1">

<DataGrid.Columns>

<DataGridTextColumn Header="Участник" Binding="{Binding FIO}" Width="4\*"></DataGridTextColumn>

<DataGridTextColumn Header="Песня" Binding="{Binding Song}" Width="4\*"></DataGridTextColumn>

<DataGridTextColumn Header="Кол-во голосов" Binding="{Binding Golosa}" Width="\*"></DataGridTextColumn>

</DataGrid.Columns>

</DataGrid>

<Button x:Name="goRight" Content=">" Grid.Column="2" Height="70" FontSize="40" Click="changeConcourse\_Click"></Button>

</Grid>

</Grid>

</Grid>

</Page>

C#:

// Обработчик изменения выбранного элемента в ComboBox для направления конкурса

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

{

Filter();

}

// Обработчик нажатия на CheckBox для отображения только победителей

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

{

Filter();

}

// Метод для применения фильтров к списку участников конкурса

private void Filter()

{

// Получение списка участников с типом "Участник"

ConcourseParticipants = dbContext.Concourse\_User.Where(rec => rec.User.UserType.name == "Участник").ToList();

// Фильтрация по выбранному направлению конкурса, если выбрано

if (directionComboBox.SelectedItem != null)

{

Concourse\_Direction concourseDirection = directionComboBox.SelectedItem as Concourse\_Direction;

ConcourseParticipants = FilterByDirection(concourseDirection, ConcourseParticipants);

}

// Фильтрация только по победителям, если выбрано

if (onlyWinners.IsChecked == true)

{

ConcourseParticipants = FilterOnlyWinners(ConcourseParticipants);

}

// Преобразование и установка источника данных для отображения

ItemsSource = ConcourseParticipants.Select(concourseUser => new UsersForVotingViewModel()

{

FIO = concourseUser.User.l\_name + " " + concourseUser.User.f\_name + " " + concourseUser.User.patronymic,

Song = concourseUser.song,

Golosa = concourseUser.golosCount,

}).OrderByDescending(cu => cu.Golosa).ToList();

// Установка источника данных для ListView

participantsListView.ItemsSource = ItemsSource;

}

// Метод для фильтрации по выбранному направлению конкурса

private List<Concourse\_User> FilterByDirection(Concourse\_Direction concourseDirection, List<Concourse\_User> dataToFilter)

{

return dataToFilter.Where(rec => rec.concourse\_direction\_id == concourseDirection.concourse\_direction\_id).ToList();

}

// Метод для фильтрации только по победителям

private List<Concourse\_User> FilterOnlyWinners(List<Concourse\_User> dataToFilter)

{

return dataToFilter.Where(rec => rec.WinnerList.Count != 0).ToList();

}

// Метод для фильтрации по конкретному конкурсу

private List<Concourse\_User> FilterByConcourse(List<Concourse\_User> dataToFilter, int concourseId)

{

return dataToFilter.Where(rec => rec.Concourse\_Direction.concourse\_id == concourseId).ToList();

}

// Обработчик нажатия кнопки сброса фильтров

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

{

// Получение исходного списка участников

GetParticipants();

// Фильтрация по выбранному конкурсу

ConcourseParticipants = FilterByConcourse(ConcourseParticipants, CurrentConcourse.Value.concourse\_id);

// Установка источника данных

SetItemsSource();

// Сброс всех фильтров

ClearFilters();

}

// Метод для сброса фильтров

private void ClearFilters()

{

// Отключение временно обработчика изменения выбранного элемента в ComboBox

directionComboBox.SelectionChanged -= directionComboBox\_SelectionChanged;

// Сброс выбранного направления

directionComboBox.SelectedItem = null;

// Включение обработчика изменения выбранного элемента в ComboBox

directionComboBox.SelectionChanged += directionComboBox\_SelectionChanged;

// Сброс флажка "Только победители"

onlyWinners.IsChecked = false;

}

# Приложение Г

Код страницы регистрации

XAML:

<Page x:Class="concourse2.Pages.OrganizerPages.OrganizerLeftPages.RegistrationPage"

d:DesignHeight="940" d:DesignWidth="1520"

Title="ConcourseDataGridPage">

<Grid Background="White">

<Grid.RowDefinitions>

<RowDefinition Height="\*"></RowDefinition>

</Grid.RowDefinitions>

<StackPanel x:Name="userData" Grid.Row="1" Margin="0 30 0 0">

<TextBlock FontFamily="Trebuchet MS" FontSize="56" Foreground="Black" Text="Регистрация" TextAlignment="Center"></TextBlock>

<WrapPanel Margin="40 70 0 0">

<StackPanel x:Name="personalData">

<WrapPanel>

<TextBlock Width="15" Style="{StaticResource MyTextBlockStyle}" Text="\*"></TextBlock>

<TextBlock Style="{StaticResource MyTextBlockStyle}" Text="Фамилия" Width="200"></TextBlock>

<TextBox x:Name="userLNameTextBox" Margin="30 0 0 0" Style="{StaticResource MyTextBoxStyle}" Text="{Binding l\_name}"></TextBox>

</WrapPanel>

<WrapPanel Margin="0 40 0 0">

<TextBlock Width="15" Text="\*" Style="{StaticResource MyTextBlockStyle}" ></TextBlock>

<TextBlock Style="{StaticResource MyTextBlockStyle}" Text="Имя" Width="200"></TextBlock>

<TextBox x:Name="userFNameTextBox" Margin="30 0 0 0" Style="{StaticResource MyTextBoxStyle}" Text="{Binding f\_name}"></TextBox>

</WrapPanel>

<WrapPanel Margin="0 40 0 0">

<TextBlock Width="15" Text="\*" Style="{StaticResource MyTextBlockStyle}" ></TextBlock>

<TextBlock Style="{StaticResource MyTextBlockStyle}" Text="Отчество" Width="200"></TextBlock>

<TextBox x:Name="userPatronymicTextBox" Margin="30 0 0 0" Style="{StaticResource MyTextBoxStyle}" Text="{Binding patronymic}"></TextBox>

</WrapPanel>

<WrapPanel Margin="0 40 0 0">

<TextBlock Width="15" Style="{StaticResource MyTextBlockStyle}" ></TextBlock>

<TextBlock Style="{StaticResource MyTextBlockStyle}" Text="Телефон" Width="200"></TextBlock>

<TextBox x:Name="userPhoneTextBox" Margin="30 0 0 0" Style="{StaticResource MyTextBoxStyle}" Text="{Binding phone}"></TextBox>

</WrapPanel>

<WrapPanel Margin="0 40 0 0">

<TextBlock Width="15" Style="{StaticResource MyTextBlockStyle}" ></TextBlock>

<TextBlock Style="{StaticResource MyTextBlockStyle}" Text="Почта" Width="200"></TextBlock>

<TextBox x:Name="userEmailTextBox" Margin="30 0 0 0" Style="{StaticResource MyTextBoxStyle}" Text="{Binding email}"></TextBox>

</WrapPanel>

<Button Margin="0 40 0 0" x:Name="registerButton" Style="{StaticResource CustomMedimButtonStyle}" Content="Зарегистрироваться" HorizontalAlignment="Left" Click="registerButton\_Click" Width="300"></Button>

</StackPanel>

<StackPanel x:Name="authData" Margin="200 0 0 0">

<WrapPanel>

<TextBlock Width="15" Text="\*" Style="{StaticResource MyTextBlockStyle}" ></TextBlock>

<TextBlock Style="{StaticResource MyTextBlockStyle}" Text="Логин" Width="200"></TextBlock>

<TextBox x:Name="userLoginTextBox" Margin="30 0 0 0" Style="{StaticResource MyTextBoxStyle}" Text="{Binding login}"></TextBox>

</WrapPanel>

<WrapPanel Margin="0 40 0 0">

<TextBlock Width="15" Text="\*" Style="{StaticResource MyTextBlockStyle}" ></TextBlock>

<TextBlock Style="{StaticResource MyTextBlockStyle}" Text="Пароль" Width="200"></TextBlock>

<TextBox x:Name="userPassTextBox" Margin="30 0 0 0" Style="{StaticResource MyTextBoxStyle}" Text="{Binding password}"></TextBox>

</WrapPanel>

</StackPanel>

</WrapPanel>

</StackPanel>

</Grid>

</Page>

C#:

// Обработчик нажатия кнопки регистрации пользователя

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

{

// Проверка валидности введенных данных пользователя

if (UserValidator.IsValid(GlobalData.DbContext, userFNameTextBox.Text, userLNameTextBox.Text, userPatronymicTextBox.Text,

userLoginTextBox.Text, userPassTextBox.Text, userPhoneTextBox.Text))

{

// Создание объекта пользователя с введенными данными

User user = new User()

{

f\_name = userFNameTextBox.Text,

l\_name = userLNameTextBox.Text,

patronymic = userPatronymicTextBox.Text,

login = userLoginTextBox.Text,

password = userPassTextBox.Text,

phone = userPhoneTextBox.Text,

email = userEmailTextBox.Text,

user\_type\_id = 1,

};

// Установка типа пользователя

user.UserType = GlobalData.DbContext.UserType.ToList().FirstOrDefault(rec => rec.user\_type\_id == 1);

// Добавление пользователя в базу данных

GlobalData.DbContext.User.Add(user);

// Сохранение изменений в базе данных

GlobalData.DbContext.SaveChanges();

// Получение пользователя из базы данных с присвоенным идентификатором

User newUser = GlobalData.DbContext.User.ToList().FirstOrDefault(rec => rec.user\_id == user.user\_id);

// Очистка полей ввода

ClearUserInput();

// Вывод сообщения об успешной регистрации

MessageBox.Show("Вы успешно зарегистрировались");

// Навигация на страницу приветствия

NavigationService.Navigate(new WelcomePage(GlobalData));

}

else

{

// Вывод сообщения об ошибке валидации

MessageBox.Show(UserValidator.Error.Message);

return;

}

}