

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ  
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
ТЕЛЕКОММУНИКАЦИЙ ИМ. ПРОФ. М.А. БОНЧ-БРУЕВИЧА»  
(СПбГУТ)

АРХАНГЕЛЬСКИЙ КОЛЛЕДЖ ТЕЛЕКОММУНИКАЦИЙ  
ИМ. Б.Л. РОЗИНГА (ФИЛИАЛ) СПбГУТ  
(АКТ (ф) СПбГУТ)

# КУРСОВОЙ ПРОЕКТ

НА ТЕМУ

РАЗРАБОТКА ПОДСИСТЕМЫ ДЛЯ УПРАВЛЕНИЯ

СОРЕВНОВАНИЯМИ ПО ТХЭКВОНДО

Л109. 25КП01. 012 ПЗ

(Обозначение документа)

МДК.02.01 Технология разработки

программного обеспечения

Студент	ИСПП-21	08.12.2025	В.А. Колосов
	(Группа)	(Подпись)	(И.О. Фамилия)
Преподаватель		09.12.2025	Ю.С. Маломан
		(Подпись)	(И.О. Фамилия)

Архангельск 2025

## СОДЕРЖАНИЕ

Перечень сокращений и обозначений .....	3
Введение .....	4
1 Анализ и разработка требований .....	6
1.1 Назначение и область применения .....	6
1.2 Постановка задачи .....	6
1.3 Выбор состава программных и технических средств .....	?
2 Проектирование программного обеспечения .....	?
2.1 Проектирование интерфейса пользователя .....	?
2.2 Разработка архитектуры программного обеспечения .....	?
2.3 Проектирование базы данных .....	?
3 Разработка и интеграция модулей программного обеспечения .....	?
3.1 Разработка программных модулей .....	?
3.2 Реализация интерфейса пользователя .....	?
3.3 Разграничение прав доступа пользователей .....	?
3.4 Экспорт и импорт данных .....	?
4 Тестирование и отладка программного обеспечения .....	?
4.1 Структурное тестирование .....	?
4.2 Функциональное тестирование .....	?
5 Инструкция по эксплуатации программного обеспечения .....	?
5.1 Установка программного обеспечения .....	?
5.2 Инструкция по работе .....	?
Заключение .....	?
Список использованных источников .....	30

## **ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ**

В настоящем курсовом проекте применяют следующие сокращения и обозначения:

АРОО – архангельская региональная общественная организация

БД – база данных

ГТФ – глобальная федерация тхэквондо

ОС – операционная система

ПК – персональный компьютер

ПО – программное обеспечение

СУБД – система управления базами данных

API – интерфейс программирования приложения

DTO – объект передачи данных

ERD – диаграмма «сущность-связь»

HTTP – протокол передачи гипертекста

IDE – интегрированная среда разработки

MVVM – архитектурный паттерн проектирования

SQL – язык структурированных запросов

SDK – комплект для разработки программного обеспечения

WPF – платформа для построения графических интерфейсов Windows

## **ВВЕДЕНИЕ**

Актуальность разрабатываемого проекта заключается в автоматизации процессов формирования турнирных сеток и учета участников турниров в спортивных организациях.

В современных условиях для эффективного управления соревнованиями важны оперативность и точность данных. Спортивные федерации и клубы сталкиваются с проблемами, связанными с учетом данных об участниках, формированием турнирных сеток и подсчетом результатов.

Разработка подсистемы для управления соревнованиями по тхэквондо позволит значительно упростить процесс организации турниров, улучшить взаимодействие с участниками и повысить общую эффективность проведения турниров.

Целью курсового проекта является разработка подсистемы, обеспечивающей возможность комплексного управления турнирами по тхэквондо, включая учет участников, формирование турнирных сеток и отслеживания результатов поединков в реальном времени.

Для достижения поставленной цели требуется решить следующие задачи:

- провести сбор и анализ требований целевой аудитории (спортивные федерации, тренеры, организаторы турниров);
- проанализировать информационные источники по предметной области;
- изучить существующие решения в области автоматизации проведения турниров;
- спроектировать архитектуру подсистемы;
- спроектировать диаграмму использования подсистемы;
- выбрать состав программных и технических средств для реализации проекта;
- спроектировать БД;

- создать БД в выбранной СУБД;
- разработать API для взаимодействия клиентских приложений с БД;
- реализовать разграничение прав доступа пользователей (организаторы, судьи);
- обеспечить защиту данных;
- разработать пользовательский интерфейс;
- реализовать функциональность регистрации участников в турнире;
- реализовать функциональность формирования турнирных сеток;
- реализовать функциональность проведения поединков;
- выполнить структурное тестирование ПО;
- выполнить функциональное тестирование ПО;
- разработать программную документацию;
- разработать эксплуатационную документацию.

В результате выполнения поставленных задач будет создана подсистема для управления соревнованиями, которая значительно упростит процесс формирования турнирных сеток и повысит скорость обработки данных участников турниров.

# **1 Анализ и разработка требований**

## **1.1 Назначение и область применения**

Основным назначением подсистемы является автоматическое формирование турнирных сеток и учет участников турнира, что позволит снизить временные затраты организаторов и судей на турнирах и минимизировать человеческие ошибки.

ПО предназначено для использования организаторами турниров по тхэквондо, судьями и участниками турниров в АРОО «Федерация тхэквондо ГТФ», а также в других спортивных федерациях и объединениях, осуществляющих проведение соревнований по тхэквондо по правилам ГТФ.

## **1.2 Постановка задачи**

Необходимо разработать оконное приложение, в котором будут реализованы следующие функциональные возможности:

- автоматическое формирование турнирных сеток;
- просмотр и редактирование данных о результатах поединков;
- просмотр и редактирование данных о турнирах;
- просмотр и редактирование данных о категориях;
- формирование отчетов по результатам соревнований в формате \*.docx;
- экспорт и импорт информации об участниках в формате \*.xlsx.

Интерфейс должен быть интуитивно понятен для пользователя.

В подсистеме должна быть обязательная авторизация пользователей.

Гость – неавторизованный пользователь, имеющий доступ к просмотру списка турниров, а также возможность зарегистрироваться в системе и выполнить вход.

Судья может просматривать список турниров, состав их участников и турнирные сетки и вносить результаты боёв в режиме реального времени.

Организатор имеет полный доступ ко всем функциям подсистемы, а также может формировать турнирные сетки, редактировать данные о категориях, участниках и турнирах.

На рисунке 1 изображена диаграмма вариантов использования подсистемы.

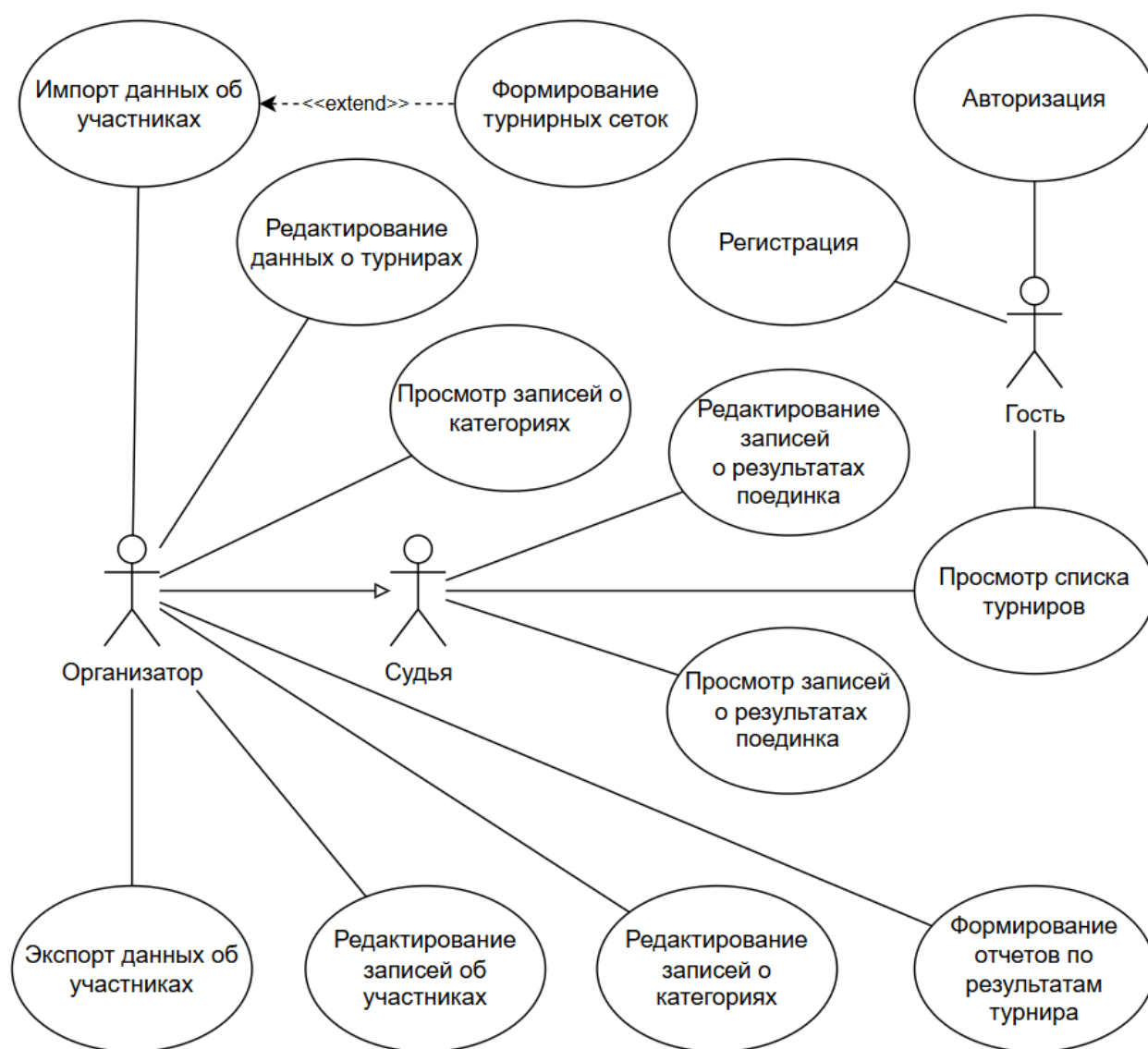


Рисунок 1 – Диаграмма вариантов использования

Работа с системой осуществляется по следующему алгоритму:

- пользователь входит в систему;

- организатор создает запись о новом турнире, заполняя всю информацию о нем;
- организатор создает записи о категориях в созданном турнире;
- система автоматически проверяет все данные и добавляет запись о турнире в общий список;
- организатор турнира импортирует данные об участниках;
- на основе данных об участниках автоматически формируются турнирные сетки;
- в ходе проведения турнира и отдельных поединков судьи и организаторы вносят информацию о результатах поединков;
- после завершения всех поединков в каждой категории система автоматически определяет победителей и призеров;
- организатор формирует и экспортирует отчеты по итогам турнира.

### **1.3 Выбор состава программных и технических средств**

Работа с оконным приложением будет осуществляться на ПК и ноутбуках с ОС Windows (Windows 10 и выше).

В качестве СУБД выбрана MySQL 8.0, так как эта СУБД обладает высокой производительностью, масштабируемостью и простотой в использовании, что позволяет эффективно обрабатывать данные о турнирах в реальном времени.

Клиентская и серверная части приложения будут разработаны на C#, так как с помощью этого языка можно эффективно создавать современные приложения с использованием технологии WPF для клиентской части и Web-API ASP.NET Core для серверной части.

Для разработки оконного приложения будет использоваться IDE Visual Studio 2022, так как эта среда предлагает удобные инструменты для работы с C#, включая инструменты для работы с Git и средства отладки.

Для функционирования системы на стороне сервера необходимы следующие программные и технические средства:

- ОС Ubuntu Server версии 24 и выше;
- сервер БД MySQL версии не ниже 8.0;
- процессор частотой 2 ГГц;
- свободная оперативная память 4 ГБ;
- свободное место на диске не менее 1 ГБ.

Для функционирования системы на стороне клиента необходимы следующие программные и технические средства:

- ОС Windows 10 и выше;
- .NET 8.0;
- процессор частотой 2 ГГц;
- свободная оперативная память 4 ГБ;
- свободное место на диске не менее 500 МБ;
- постоянное подключение к локальной сети или сети интернет.

Указанный набор программных средств обеспечит эффективное функционирование оконного приложения.

## 2 Проектирование программного обеспечения

### 2.1 Проектирование интерфейса пользователя

В рамках разработки оконного приложения создан интерфейс пользователя в виде набора wireframe при помощи инструмента draw.io. Эти визуальные представления позволяют представить структуру приложения, его основные элементы и функциональность.

Wireframe страниц «Авторизация», «Турниры», «Создание турнира», «Редактирование турнира» представлен на рисунке 2.

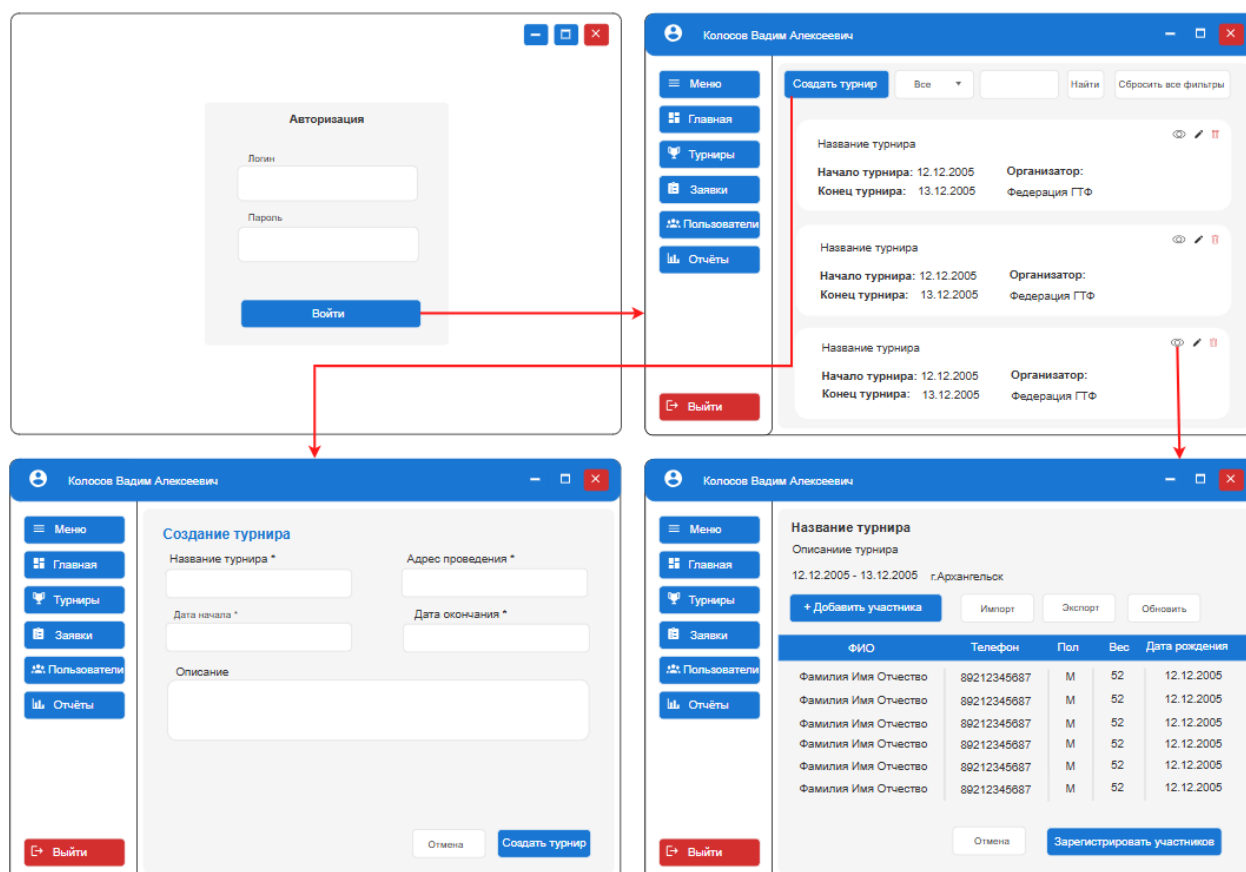


Рисунок 2 – Wireframe основных окон подсистемы

Для оконного приложения были выбраны следующие цвета:

- основной цвет: #FF1976D2;

- цвет фона: #FFF5F5F5;
- основной цвет текста: #FF333333;
- вторичный цвет текста: #FF666666.

## 2.2 Разработка архитектуры программного обеспечения

Архитектура построена на основе клиент-серверной модели и включает в себя несколько ключевых компонентов: оконное приложение, БД, API, позволяющий клиенту взаимодействовать с сервером. Диаграмма развертывания изображена на рисунке 3.

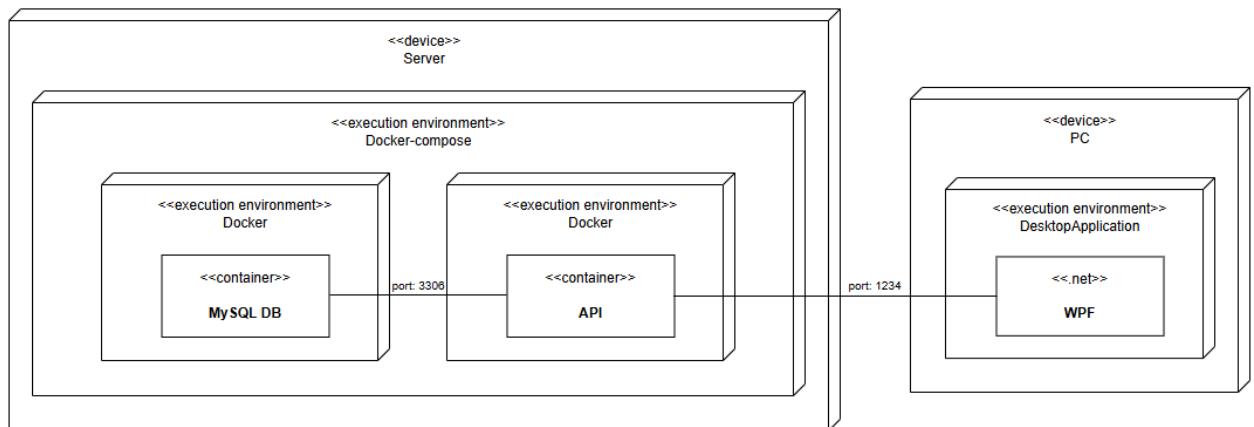


Рисунок 3 – Диаграмма развертывания

## 2.3 Проектирование базы данных

В рамках проектирования подсистемы требуется разработать БД для хранения данных о турнирах, их участниках и турнирах, турнирных сетках и пользователях.

БД спроектирована с учетом нормализации до третьей нормальной формы, что минимизирует избыточность данных и обеспечивает целостность информации.

На рисунке 4 в виде ERD показана физическая модель предметной области, созданная при помощи MySQL Workbench.

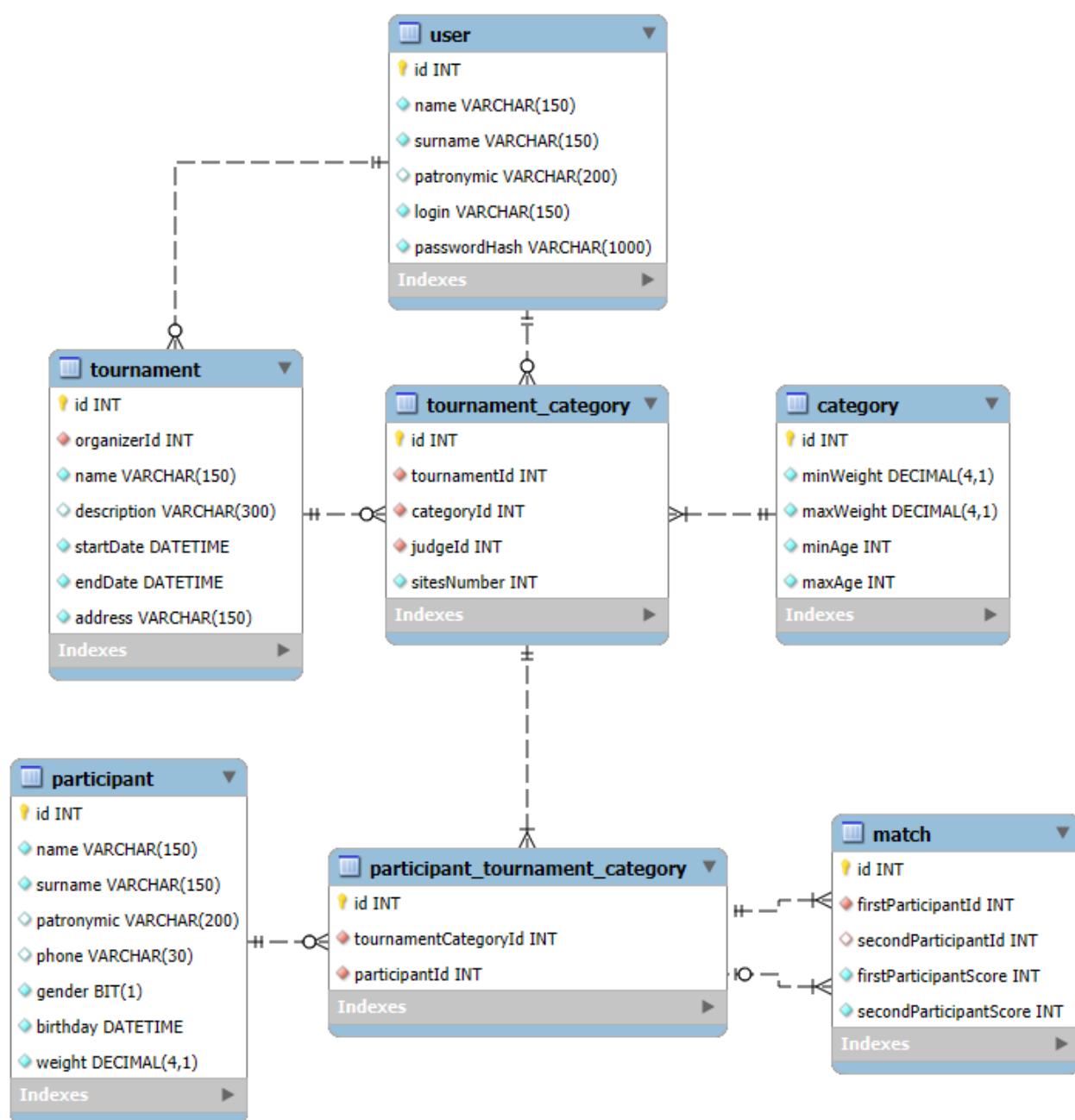


Рисунок 4 – Физическая модель БД

## 3 Разработка и интеграция модулей программного обеспечения

### 3.1 Разработка программных модулей

В ходе проектирования разработаны БД в MySQL Workbench, Web-API, оконное приложение WPF на C# в Visual Studio 2022.

Приложение разрабатывается с использованием паттерна проектирования MVVM. Для передачи данных между клиентом и сервером используются DTO, которые обеспечивают сериализацию и десериализацию объектов при сетевом взаимодействии.

Взаимодействие приложения с сервером осуществляется при помощи HTTP-запросов к Web-API с использованием библиотеки Entity Framework Core. С помощью данной библиотеки реализованы модели и контекст БД. Ответы возвращаются в виде HTTP-статуса и ссылки на созданный объект. Для выполнения HTTP-запросов использован HttpClient. Код метода создания записи о турнире в БД отправкой POST-запроса на сервер представлен листингом 1.

#### Листинг 1 – Код метода PostTournament

```
[HttpPost]
public async Task<IActionResult> PostTournament(TournamentDto
tournamentDto)
{
    //Создание нового объекта Tournament на основе данных из DTO
    var tournament = new Tournament
    {
        Id = tournamentDto.Id,
        Name = tournamentDto.Name,
        Description = tournamentDto.Description,
        StartDate = tournamentDto.StartDate,
        EndDate = tournamentDto.EndDate,
        Address = tournamentDto.Address,
        OrganizerId = tournamentDto.OrganizerId
    }
```

```

};

//Добавление турнира в контекст базы данных
context.Tournaments.Add(tournament);
//Сохранение изменений в базе данных
await context.SaveChangesAsync();
//Возврат ответа 201 Created с ссылкой на созданный ресурс
return CreatedAtAction("GetTournament", new { id =
tournamentDto.Id }, tournament);
}

```

Для получения списка турниров в приложении разработан метод LoadTournaments, код которого представлен листингом 2.

## Листинг 2 – Код метода LoadTournaments

```

[RelayCommand]
private async Task LoadTournaments()
{
    //Установка флага загрузки в true для отображения индикатора
    загрузки
    IsLoading = true;

    try
    {
        // Асинхронное получение списка турниров из сервиса
        var tournaments = await
        _tournamentService.GetAllAsync();
        // Проверка, что полученные данные не null
        if (tournaments is not null)
        {
            _allTournaments.Clear();
            // Добавление каждого турнира в коллекцию с
            проверкой на null
            foreach (var tournament in tournaments)
            {
                if (tournament is not null)
                    _allTournaments.Add(tournament);
            }
            // Применение фильтров к загруженным данным
            ApplyFilters();
        }
        catch (Exception ex)
        {
            // Обработка ошибок с выводом сообщения пользователю
            MessageBox.Show($"Ошибка загрузки турниров:
            {ex.Message}", "Ошибка");
        }
    }
}

```

```

    }
    finally
    {
        // Сброс флага загрузки независимо от результата
        // выполнения
        IsLoading = false;
    }
}

```

Изменение информации в БД осуществляется посредством Web-API приложения, код метода изменения информации о турнире по идентификатору представлен листингом 3.

### Листинг 3 – Код функции PutTournament

```

[HttpPut("{id}")]
public async Task<IActionResult> PutTournament(int id,
TournamentDto tournamentDto)
{
    //Проверка соответствия идентификатора в URL и в теле
    //запроса
    if (id != tournamentDto.Id)
        return BadRequest();

    try
    {
        //Создание объекта Tournament на основе данных из DTO
        var tournament = new Tournament
        {
            Id = tournamentDto.Id,
            Name = tournamentDto.Name,
            Description = tournamentDto.Description,
            StartDate = tournamentDto.StartDate,
            EndDate = tournamentDto.EndDate,
            Address = tournamentDto.Address,
            OrganizerId = tournamentDto.OrganizerId
        };

        //Установка состояния сущности как измененной для
        //обновления в БД
        context.Entry(tournament).State = EntityState.Modified;
        //Сохранение изменений в базе данных
        await context.SaveChangesAsync();
    }
    catch (DbUpdateConcurrencyException)
    {
        // Обработка ошибки параллельного доступа к данным
        if (!TournamentExists(id))
    }

```

```

        return NotFound(); // Турнир не найден
        return BadRequest();
    }

    // Возврат статуса 204 No Content при успешном обновлении
    return NoContent();
}

```

### 3.2 Реализация интерфейса пользователя

Интерфейс приложения реализован с использованием постраничной навигации в статичном главном окне. В приложении разработаны различные элементы управления, стили и карточки для упрощения работы. Навигация в приложении осуществляется с помощью бокового меню, которое обеспечивает переход между страницами.

Для отображения информации о турнире разработана карточка, вид которой представлен на рисунке 5.

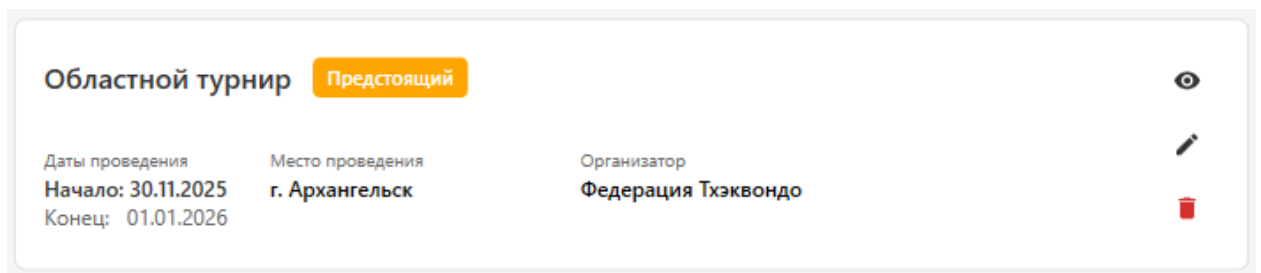


Рисунок 5 – Tournament Manager. Вид карточки турнира

Фрагмент кода вида карточки турнира, формирующий кнопки управления, представлен листингом 4.

#### Листинг 4 – Фрагмент кода вида карточки турнира

```

<StackPanel Grid.Column="1"
            HorizontalAlignment="Right"
            VerticalAlignment="Top">
    <!--Код для просмотра деталей турнира -->

```

```

        <Button Command="{Binding
DataContext.ViewTournamentDetailsCommand,
RelativeSource={RelativeSource AncestorType=ItemsControl}}"
            CommandParameter="{Binding}"
            Style="{StaticResource IconButtonStyle}">
            <!-- Иконка "глаз" для визуального обозначения просмотра
-->
            <materialDesign:PackIcon Kind="Eye"/>
        </Button>
        <!--Кнопка для редактирования турнира -->
        <Button Command="{Binding DataContext.EditTournamentCommand,
RelativeSource={RelativeSource AncestorType=ItemsControl}}"
            CommandParameter="{Binding}"
            Style="{StaticResource IconButtonStyle}">
            <!-- Иконка "карандаш" для визуального обозначения
редактирования -->
            <materialDesign:PackIcon Kind="Pencil"/>
        </Button>

        <!-- Кнопка для удаления турнира -->
        <Button Command="{Binding
DataContext.DeleteTournamentCommand,
RelativeSource={RelativeSource AncestorType=ItemsControl}}"
            CommandParameter="{Binding}"
            Style="{StaticResource IconButtonStyle}"
            Foreground="{StaticResource ErrorBrush}">
            <!-- Иконка "корзина" для визуального обозначения
удаления -->
            <materialDesign:PackIcon Kind="Delete"/>
        </Button>
</StackPanel>

```

В приложении также разработаны стили для различных элементов интерфейса. Фрагмент кода стиля для кнопки в виде иконки представлен листингом 5.

#### Листинг 5 – Фрагмент кода стиля для кнопки в виде иконки

```

<Style x:Key="IconButtonStyle" TargetType="Button">
    <!-- Свойства высоты и ширины элемента -->
    <Setter Property="Width" Value="36"/>
    <Setter Property="Height" Value="36"/>
    <!-- Свойство отступов элемента -->
    <Setter Property="Padding" Value="8"/>
    <!-- Свойства цвета фона элемента -->
    <Setter Property="Background" Value="Transparent"/>
    <Setter Property="BorderBrush" Value="Transparent"/>
    <!-- Свойство толщины границы элемента -->

```

```

        <Setter Property="BorderThickness" Value="0"/>
        <!-- Свойство цвета шрифта элемента -->
        <Setter Property="Foreground" Value="{StaticResource
TextPrimaryBrush}"/>
        <!-- Свойство определения типа курсора при наведении на
элемент -->
        <Setter Property="Cursor" Value="Hand"/>
        <!-- Свойство вертикального выравнивания контента внутри
элемента -->
        <Setter Property="VerticalContentAlignment" Value="Top"/>
        <Setter Property="Template">
            <Setter.Value>
                <!-- Шаблон кнопки -->
                <ControlTemplate TargetType="Button">
                    <Border x:Name="border"
                        Background="{TemplateBinding
Background}"
                        BorderBrush="{TemplateBinding
BorderBrush}"
                        BorderThickness="{TemplateBinding
BorderThickness}"
                        CornerRadius="18"
                        Width="{TemplateBinding Width}"
                        Height="{TemplateBinding Height}">
                        <ContentPresenter x:Name="contentPresenter"
Content="{TemplateBinding Content}"
ContentTemplate="{TemplateBinding ContentTemplate}"
HorizontalAlignment="Center" VerticalAlignment="Center"/>
                    </Border>
                    <!-- Триггеры для визуальной обратной связи
кнопки -->
                    <ControlTemplate.Triggers>
                        <!-- Изменение фона при наведении курсора -->
                        <Trigger Property="IsMouseOver"
Value="True">
                            <Setter TargetName="border"
Property="Background" Value="{StaticResource SecondaryBrush}"/>
                        </Trigger>
                        <!-- Изменение фона при нажатии кнопки -->
                        <Trigger Property="IsPressed" Value="True">
                            <Setter TargetName="border"
Property="Background" Value="{StaticResource BorderBrush}"/>
                        </Trigger>
                        <!-- Стил для неактивного состояния -->
                        <Trigger Property="IsEnabled" Value="False">
                            <Setter Property="Foreground"
Value="{StaticResource TextHintBrush}"/>
                        </Trigger>
                    </ControlTemplate.Triggers>
                </ControlTemplate>
            </Setter.Value>
        </Setter>
    </Style>

```

### 3.3 Разграничение прав доступа пользователей

Разграничение прав доступа реализовано разделением организаторов и судей по разным таблицам БД. Права организатора у пользователя появляются при значении True в поле IsOrganizer.

Пример кода разграничения прав пользователей приложения представлен листингом 6.

Листинг 6 – Код разграничения прав пользователей приложения

```
private void InitializeNavigation()
{
    //Инициализация коллекции элементов меню
    MenuItems = new ObservableCollection<MenuItem>();

    //Проверка роли пользователя (организатор или обычный
    пользователь)
    if (CurrentUser.IsOrganizer)
    {
        //Добавление пунктов меню для организатора
        MenuItems.Add(new MenuItem("Заявки", new RelayCommand(()
=> Navigate("Applications")), "ClipboardList"));
        MenuItems.Add(new MenuItem("Пользователи", new
RelayCommand(() => Navigate("Users")), "AccountGroup"));
        MenuItems.Add(new MenuItem("Отчёты", new RelayCommand(()
=> Navigate("Reports")), "ChartBar"));
    }
    else
    {
        //Добавление пунктов меню для обычного пользователя
        MenuItems.Add(new MenuItem("Мои площадки", new
RelayCommand(() => Navigate("MyMatches")), "Stadium"));
        MenuItems.Add(new MenuItem("Судьи", new RelayCommand(()
=> Navigate("Judges")), "Gavel"));
    }
}
```

### 3.4 Экспорт и импорт данных

Для автоматизации процесса получения данных об участниках из турнира в приложении реализован экспорт файла в формате xlsx. Код метода экспорта данных об участниках представлен листингом 7.

## Листинг 7 – Код метода ExportParticipants

```
[RelayCommand]
private async Task ExportParticipants()
{
    //Проверка прав доступа: только организаторы могут
    экспортировать участников
    if (!IsOrganizer)
    {
        MessageBox.Show("Доступ запрещен. Только организаторы
могут экспортировать участников.", "Ошибка доступа");
        return;
    }

    try
    {
        //Проверка наличия участников для экспорта
        if (!Participants.Any())
        {
            MessageBox.Show("Нет участников для экспорта",
"Экспорт");
            return;
        }

        //Настройка диалога сохранения файла
        var saveFileDialog = new SaveFileDialog
        {
            Filter = "Excel files (*.xlsx)|*.xlsx|All files
(*.*)|*.*",
            Title = "Сохранить файл с участниками",
            FileName = $"Участники_турнира_{DateTime.Now:yyyy-
MM-dd}.xlsx"
        };

        //Если пользователь выбрал место сохранения и подтвердил
        if (saveFileDialog.ShowDialog() == true)
        {
            //Включение индикатора загрузки
            IsLoading = true;

            //Асинхронный экспорт данных в Excel файл
            await Task.Run(() =>
ExportToExcel(saveFileDialog.FileName));

            //Уведомление пользователя об успешном экспорте
            MessageBox.Show($"Участники успешно экспортированы в
файл: {saveFileDialog.FileName}", "Экспорт завершен");
        }
    }
    catch (Exception ex)
    {
        //Обработка ошибок при экспорте
    }
}
```

```

        MessageBox.Show($"Ошибка при экспорте участников:
{ex.Message}", "Ошибка экспорта");
    }
    finally
    {
        //Выключение индикатора загрузки в любом случае
        IsLoading = false;
    }
}

```

Для автоматизации добавления данных об участниках в турнир реализован импорт из файла в формате xlsx. Фрагмент кода метода импорта данных об участниках представлен листингом 8.

#### Листинг 8 – Фрагмент кода метода ImportParticipants

```

//Настройка диалога выбора файла для импорта
var openFileDialog = new OpenFileDialog
{
    Filter = "Excel files|*.xlsx;*.xls|All files|*.*",
    Title = "Выберите файл для импорта участников"
};
//Если пользователь выбрал файл и подтвердил выбор
if (openFileDialog.ShowDialog() == true)
{
    // Включение индикатора загрузки
    IsLoading = true;

    //Парсинг Excel-файла и получение списка участников
    var importedParticipants = await
    ParseExcelFile(openFileDialog.FileName);
    //Если участники были успешно импортированы
    if (importedParticipants.Any())
    {
        //Добавление каждого участника в основную коллекцию
        foreach (var participant in importedParticipants)
            Participants.Add(participant);
        //Установка флага наличия несохраненных изменений
        HasUnsavedChanges = true;
        MessageBox.Show($"Успешно импортировано
{importedParticipants.Count} участников", "Импорт завершен");
    }
    else
    {
        MessageBox.Show("Не удалось импортировать участников из
файла", "Ошибка импорта");
    }
}

```

## 4 Тестирование и отладка программного обеспечения

### 4.1 Структурное тестирование

Во время проектирования проведено тестирование методом белого ящика метода редактирования информации о категории в Web-API, результаты которого представлены в таблице 1 [1].

Таблица 1 – Результаты тестирования метода редактирования информации о категории

Действие	Ожидаемый результат	Фактический результат
1) Проверить, что категория с переданным id существует 2) Выполнить PUT запрос в API с телом: { " id": 0, "minWeight": 50, "maxWeight": 100, "minAge": 15, "maxAge": 20 }	Получение HTTP-ответа 404 Not Found. В ответе получены время, тип ошибки и информация о том, что категория не найдена	Совпадает с ожидаемым
1) Проверить, что категория с переданным id существует 2) Выполнить PUT запрос в API с телом: { " id": 20, "minWeight": 350, "maxWeight": 400, "minAge": 100, "maxAge": 100 }	Получение HTTP-ответа 400 Undocumented. В ответе получены время, тип ошибки и информация о том, что превышено значение в полях minWeight и maxWeight	Совпадает с ожидаемым
1) Проверить, что категория с переданным id существует 2) Выполнить PUT запрос в API с телом: { " id": 20, "minWeight": 75, "maxWeight": 85, "minAge": 18, "maxAge": 25 }	Получения HTTP-ответа 204 Undocumented. В ответе получено время обновления информации о категории	Совпадает с ожидаемым

## 4.2 Функциональное тестирование

Во время разработки проведено функциональное тестирование приложения методом черного ящика, результаты тестирования представлены в таблице 2.

Таблица 2 – Результаты тестирования приложения методом черного ящика

Действие	Ожидаемый результат	Фактический результат
Нажать на кнопку трехстрочного меню	Скрытие меню навигации, сккрытие текста кнопок перехода между страницами	Совпадает с ожидаемым
Нажать на кнопку «Турниры» в навигационном меню	Отображение списка всех турниров в рабочей области с элементами управления и фильтрации	Совпадает с ожидаемым
Нажать на кнопку «Создать турнир» на странице со списком турниров	Переход на экран «Создание турнира»	Совпадает с ожидаемым
Нажать на кнопку «Создать турнир» на странице создания турниров	Отображение в модальном окне ошибки валидации, т.к. поля не заполнены	Совпадает с ожидаемым
Нажать на кнопку «Отмена» на странице создания турниров	Переход на страницу со списком турниров	Совпадает с ожидаемым
Нажать на кнопку с иконкой «глаз» на карточке турнира	Переход на форму «Детали турнира»	Совпадает с ожидаемым
Выделить определенный диапазон участников в таблице, нажать кнопку «Удалить», подтвердить удаление	Отображение модального окна с подтверждением удаления, удаление диапазона участников с таблицы	Совпадает с ожидаемым
Продублировать в таблице информацию о нескольких участниках, нажать кнопку «Убрать дубликаты»	Отображение модального окна с информацией об удаленных дубликатах	Совпадает с ожидаемым
Нажать на кнопку «Управление категориями»	Открытие окна взаимодействия с категориями турнира	Совпадает с ожидаемым
Выбрать категорию в списке со всеми категориями, нажать на кнопку «Корзина», подтвердить удаление	Категория удалена из списка	Совпадает с ожидаемым

По результатам тестирования можно сделать вывод, что разработанное приложение работает корректно и согласно ожиданиям.

## **5 Инструкция по эксплуатации программного обеспечения**

Для функционирования подсистемы на стороне сервера достаточны следующие программные и технические средства:

- ОС Ubuntu Server версии 24 и выше;
- процессор частотой 2 ГГц;
- свободная оперативная память 4 ГБ;
- свободное место на диске не менее 1 ГБ;
- дополнительные компоненты; Docker, docker-compose, MySQL 8.0, .Net SDK версии 8.0 и выше,

### **5.1 Установка программного обеспечения**

### **5.2 Инструкция по работе**





## **ЗАКЛЮЧЕНИЕ**

Объем 1 полная страница

Цели достигнуты, задачи выполнены, итог

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Бек, К. Экстремальное программирование: разработка через тестирование. – Санкт-Петербург : Питер, 2021. – 224 с. – URL: <https://ibooks.ru/bookshelf/376974/reading> (дата обращения: 01.11.2025). – Режим доступа: для зарегистрир. пользователей. – Текст : электронный.
2. Гагарина, Л. Г. Технология разработки программного обеспечения : учебное пособие / Л. Г. Гагарина, Е. В. Кокорева, Б. Д. Сидорова-Виснадул ; под ред. Л. Г. Гагариной. – Москва : ФОРУМ : ИНФРА-М, 2025. – 400 с. – URL: <https://znanium.ru/catalog/product/2178802> (дата обращения: 12.11.2025). – Режим доступа: по подписке. – Текст : электронный.
3. Мартишин, С. А. Базы данных. Практическое применение СУБД SQL и NoSQL-типа для проектирования информационных систем : учебное пособие / С. А. Мартишин, В. Л. Симонов, М. В. Храпченко. – Москва : ФОРУМ : ИНФРА-М, 2024. – 368 с. – URL: <https://znanium.ru/catalog/product/2096940> (дата обращения: 23.11.2025). – Режим доступа: по подписке. – Текст : электронный.
4. Тидвелл, Д. Разработка интерфейсов. Паттерны проектирования. 3-е изд. – Санкт-Петербург : Питер, 2022. – 560 с. – URL: <https://ibooks.ru/bookshelf/386796/reading> (дата обращения: 23.11.2025). – Режим доступа: для зарегистрир. пользователей. – Текст : электронный.
5. Федорова, Г. Н. Разработка, внедрение и адаптация программного обеспечения отраслевой направленности : учебное пособие. – Москва : КУРС : ИНФРА-М, 2024. – 336 с. – URL: <https://znanium.ru/catalog/product/2083407> (дата обращения: 24.11.2025). – Режим доступа: по подписке. – Текст : электронный.