Центросоюз РФ

Нижегородский областной союз потребительских обществ

«Нижегородский экономико-технологический колледж»

ДНЕВНИК ПРОИЗВОДСТВЕННОЙ ПРАКТИКИ

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Индекс и название профессионального модуля

Специальность\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Код и наименование специальности

Обучающегося(йся) \_\_\_\_\_\_\_\_\_ курса \_\_\_\_\_\_\_\_ группы

форма обучения очная

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Фамилия, имя, отчество обучающегося

Место практики \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Название организации

Срок практики с \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2023 г. по \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2023 г.

Арзамас, 2023

Содержание дневника

|  |  |  |
| --- | --- | --- |
| Дата | Описание выполненной работы | Оценка и подпись руководителя практики |
| 1 | 2 | 3 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Содержание объема выполненных работ подтверждаю.

Руководитель практики от предприятия: \_\_\_\_\_\_\_\_\_\_\_\_\_ /\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/

подпись Ф.И.О.

Центросоюз РФ

Нижегородский областной союз потребительских обществ

«Нижегородский экономико-технологический колледж»

ОТЧЕТ О ПРОИЗВОДСТВЕННОЙ ПРАКТИКЕ

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Индекс и название профессионального модуля

Специальность\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Код и наименование специальности

Обучающегося(йся) \_\_\_\_\_\_\_\_\_ курса \_\_\_\_\_\_\_\_ группы

форма обучения очная

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Фамилия, имя, отчество обучающегося

Место практики \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Название организации

Срок практики с «\_\_\_»\_\_\_\_\_\_\_\_\_\_2023 г. по «\_\_\_»\_\_\_\_\_\_\_\_\_\_2023 г.

Руководители практики

от организации \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

должность подпись ФИО

от колледжа \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

должность подпись ФИО

Итоговая оценка по практике \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Арзамас, 2023

СОДЕРЖАНИЕ

1. ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ 3
2. РАЗРАБОТКА КОДА ИНФОРМАЦИОННОЙ СИСТЕМЫ
3. ТЕСТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ

ВЫВОДЫ 25

ПРИЛОЖЕНИЯ

1. ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ

Образовательное учреждение столкнулось с необходимостью обновления программы по оплате проживающих в общежитии и обучающихся в колледже. Данная глава посвящена проектированию информационной системы NETKOplata, разработанной специально для колледжа НЭТК, с целью повышения эффективности управления и анализа данных.

Процесс проектирования начинается с внимательного изучения предоставленных материалов. Этот этап позволяет получить полное представление о потребностях и особенностях колледжа в автоматизации контроля данными по оплате проживающих и обучающихся.

NETKOplata разработана с целью обеспечить эффективное управление и анализ в образовательном учреждении. Система позволяет легко хранить и управлять всей необходимой документацией, связанной с плательщиками, а также автоматизировать процессы учета и анализа данных. NETKOplata способствует повышению прозрачности и оперативности в принятии решений, обеспечивая более эффективное управление данными.

Следующим шагом является разработка модели информационной системы, начиная с UseCase-диаграммы (рисунок 1). В информационной системе предусмотрена одна роль - Бухгалтер. Бухгалтер имеет возможность просматривать список студентов, проживающих в общежитии, а также добавлять, редактировать и удалять данные из базы данных. Также бухгалтер управляет списком групп, устанавливает категории плательщиков и определенную сумму оплаты за каждый месяц проживания.

Проектирование информационной системы NETKOplata представляет собой важный шаг в совершенствовании системы управления данными в колледже НЭТК. Реализация данной системы позволит значительно улучшить прозрачность, оперативность и эффективность процессов учета и анализа данных, что в конечном итоге способствует более эффективному управлению образовательным процессом.

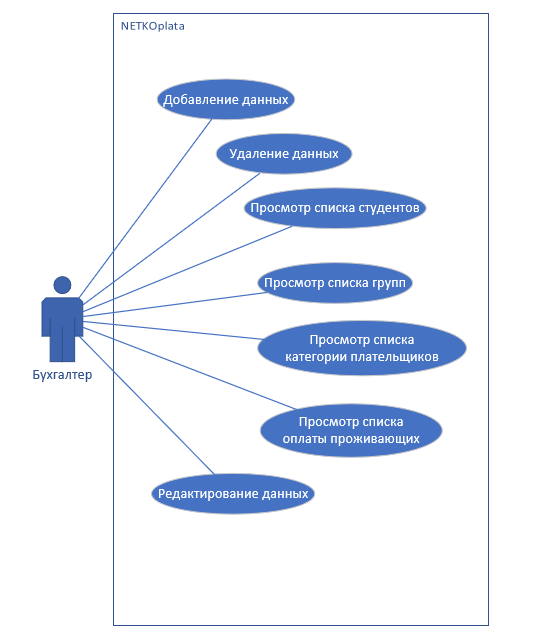


Рисунок 1 – UseCase диаграмма

Следующим важным этапом в проектировании информационной системы NETKOplata для колледжа НЭТК является разработка диаграммы Activity. Эта диаграмма дополнит предыдущую UseCase диаграмму, предоставив более детальное представление о взаимодействии компонентов системы в рамках конкретных сценариев использования (рисунок 2).

Цель данной диаграммы заключается в том, чтобы проиллюстрировать последовательность действий и взаимодействие между различными компонентами информационной системы NETKOplata в рамках конкретных сценариев использования.

Разработка диаграммы Activity представляет собой важный этап в проектировании информационной системы NETKOplata. Эта диаграмма будет дополнять UseCase диаграмму, обеспечивая более глубокое понимание взаимодействия компонентов системы в рамках конкретных сценариев использования. Это позволит более эффективно управлять процессами и обеспечивать высокую функциональность системы.

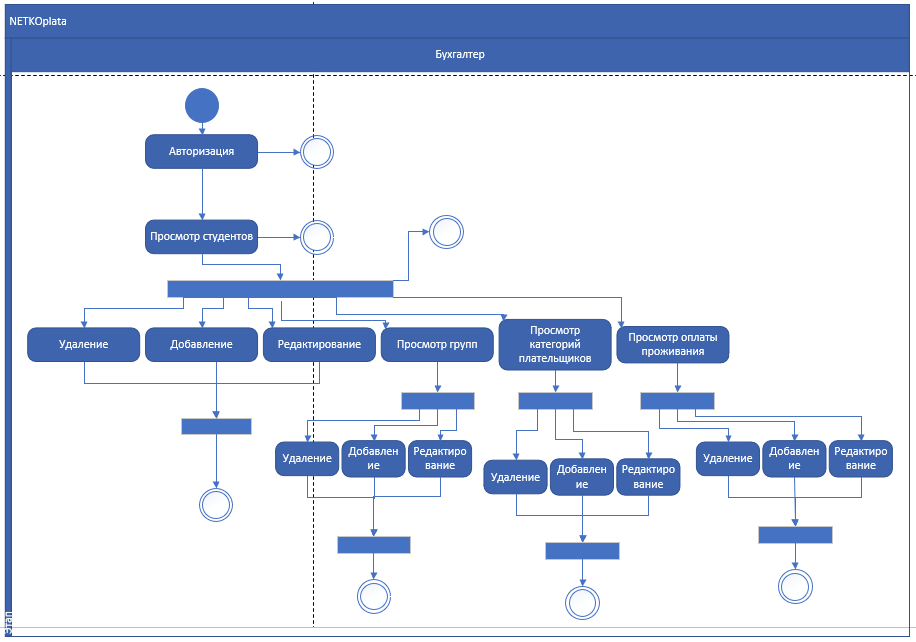


Рисунок 2 – Activity диаграмма

Третьим ключевым элементом в проектировании информационной системы NETKOplata для колледжа НЭТК является диаграмма сущность-связь (ERD). ERD предоставляет визуальное представление структуры базы данных и связей между различными сущностями, которые система будет хранить.

Диаграмма сущность-связь включает следующие ключевые концепции:

1. Сущности (Entities): основные объекты, о которых будет храниться информация. В контексте NETKOplata, "Студент" и "Группа" могут быть примерами сущностей.
2. Связи (Relationships): определяют взаимодействие между сущностями. Например, "Студент обучается в Группе" представляет собой связь между сущностями "Студент" и "Группа".
3. Атрибуты (Attributes): характеристики сущностей. Например, у сущности "Студент" могут быть атрибуты "ФИО" и "Группа".

Процесс разработки ERD включает в себя:

* Определение основных объектов и их характеристик.
* Определение взаимодействия между сущностями.

Определение, сколько сущностей одного типа может связаться с сущностью другого типа.

В случае NETKOplata, ERD может включать сущности " Студент" и " Группа", их атрибуты (например, данные о сотруднике и характеристики документа) и связи, отражающие взаимодействие между ними.

Разработка ERD является важным этапом проектирования NETKOplata. Она позволяет четко определить структуру базы данных, учитывая сущности, атрибуты и связи между ними. Создание эффективной и надежной информационной системы, ориентированной на управление данными в колледже НЭТК, начинается с ясного понимания этой структуры (рисунок 3).

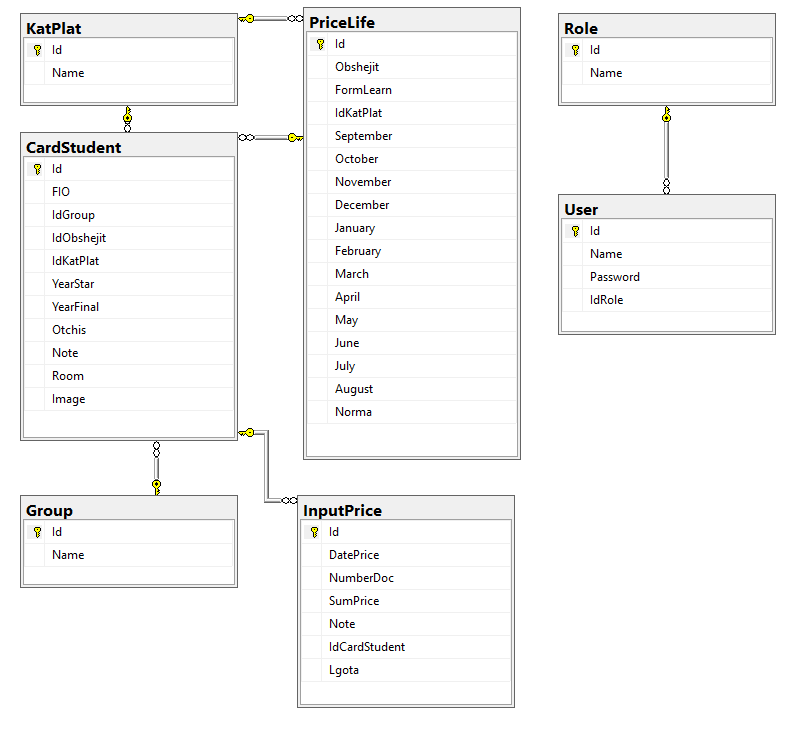


Рисунок 3 – ER – диаграмма

В ходе проектирования информационной системы NETKOplata для колледжа НЭТК были выполнены ключевые этапы, начиная от анализа проектной и технической документации и заканчивая разработкой макетов пользовательского интерфейса.

Тщательное изучение проектной и технической документации позволило точно определить особенности работы колледжа и выявить требования к системе NETKOplata.

На основе анализа документации были выработаны функциональности, соответствующие стандартам и нормативам образовательной сферы.

Создание диаграмм UseCase, Activity и ERD позволило визуализировать взаимодействие пользователей с системой и структуру базы данных, обеспечивая основу для эффективного управления данными.

Разработка интуитивно понятных и удобных в использовании макетов UI была направлена на удовлетворение потребностей различных пользовательских категорий, с особенным акцентом на комфортное взаимодействие бухгалтеров с системой.

Полученные результаты создают прочный фундамент для NETKOplata, обеспечивая эффективное управление данными о персонале и документах в колледже НЭТК. Комплексный подход к проектированию системы гарантирует соответствие ее функциональности и интерфейса высоким стандартам в образовательной сфере.

1. РАЗРАБОТКА КОДА ИНФОРМАЦИОННОЙ СИСТЕМЫ

Перед началом разработки информационной системы NETKOplata для колледжа НЭТК были проведены тщательные исследования существующих методов и инструментов. В результате анализа был сформирован оптимальный технологический стек, который отвечает требованиям проекта.

C# был выбран в качестве основного языка программирования. Этот объектно-ориентированный язык, ориентированный на компоненты, предоставляет конструкции для непосредственной поддержки концепции работы с компонентами. C# идеально подходит для создания программных компонентов и обладает богатой функциональностью, соответствующей современным требованиям по разработке ПО.

Microsoft SQL Server выбран для работы с базой данных. Эта система обеспечивает функционал по хранению и извлечению данных, предоставляя возможность обработки запросов из других программных приложений. Различные версии SQL Server адаптированы для различных аудиторий и рабочих нагрузок, что делает ее идеальным выбором для широкого спектра приложений.

Для управления базой данных SQL Server предпочтение отдано SQL Server Management Studio (SSMS) - интегрированной среде, обеспечивающей графические инструменты и редакторы скриптов для эффективного взаимодействия с SQL Server. SSMS поддерживает разработчиков и администраторов баз данных всех профессиональных уровней.

Visual Studio выбрана в качестве платформы для написания, отладки, сборки кода и публикации приложений. Эта интегрированная среда разработки предоставляет обширный инструментарий, включая компиляторы, средства автозавершения кода, графические конструкторы и другие функции, облегчающие процесс разработки. Важной частью является также .NET Framework, предоставляющий среду выполнения и библиотеки классов для языка C#.

Для взаимодействия с базой данных и управления объектами предметной области будет использован Entity Framework. Этот фреймворк обеспечивает удобный способ работы с данными, используя объектно-ориентированный подход.

WPF выбрана в качестве платформы пользовательского интерфейса. WPF, не завися от разрешения, использует векторный механизм визуализации, предоставляя преимущества современного графического оборудования. Этот инструмент предоставляет обширный набор функций для разработки приложений, включая язык XAML, элементы управления, привязку к данным, графику, анимацию, стили, шаблоны и многое другое.

Выбранный технологический стек, объединяющий C#, .NET Framework, Entity Framework, Microsoft SQL Server и WPF, обеспечивает эффективное, современное и масштабируемое решение для создания информационной системы NETKOplata. Этот стек инструментов обеспечивает не только высокую производительность, но и удобство разработки, обеспечивая возможность создания полнофункционального приложения, соответствующего требованиям колледжа НЭТК.

В процессе инициации проектирования информационной системы "NETKOplata" используется SQL Server Management Studio (SSMS). На данном этапе формируется база данных с разработкой диаграммы, включающей ключевые сущности, их атрибуты и установку взаимосвязей. Создается база данных "NETKOplata" в SSMS, где определяются типы данных, первичные и внешние ключи. Завершающим шагом является настройка индексаций для оптимизации выполнения запросов и повышения производительности базы данных, получая готовую структуру базы данных (рисунок 4).

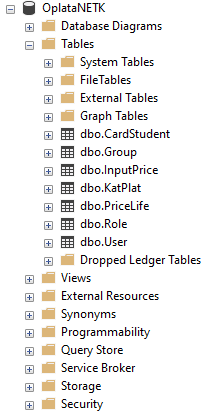


Рисунок 4 – База данных в SSMS

После успешного создания базы данных "NETKOplata" наступает этап заполнения тестовыми данными. Этот шаг критичен для проверки функциональности разрабатываемого приложения. Обеспечивая базу данных необходимым для тестирования набором данных, мы гарантируем эффективное тестирование каждого аспекта системы.

Теперь, когда база данных содержит необходимые тестовые данные, мы переходим к следующему важному этапу - подключению базы данных к информационной системе "NETKOplata". Для реализации данной задачи мы используем ADO модель.

ADO (ActiveX Data Objects) представляет собой объектно-ориентированный интерфейс для технологии доступа к данным OLE DB. Технология ADO обеспечивает ключевые возможности для построения клиент/серверных и Web-приложений. Кроме того, она поддерживает функции Remote Data Service (RDS), позволяя перемещать данные между сервером и клиентским приложением или веб-страницей, манипулировать данными "на стороне клиента" и возвращать обновленные данные серверу.

Процесс подключения базы данных к информационной системе, следующий:

1. В структуру проекта информационной системы добавляем объект «Модель ADO .NET EDM».
2. Задаем модели название "NETKOplataModel", обеспечивая ясность и структурированность проекта.

Этот шаг является важным моментом интеграции, где мы гарантируем, что разрабатываемая информационная система успешно взаимодействует с созданной базой данных, обеспечивая стабильную и эффективную работу системы "NETKOplata".

После выбора необходимого элемента и нажатия на кнопку добавления, мы переходим к следующему этапу - открытию мастера моделей EDM (Entity Data Model). Здесь предстоит выбрать содержание модели, определяя основные аспекты для дальнейшей разработки.

В данном проекте мы приняли решение использовать "Конструктор EF из базы данных" в мастере моделей EDM. Этот выбор обусловлен необходимостью создания модели, основанной на структуре базы данных, что в свою очередь облегчит интеграцию и обеспечит соответствие модели реальным данным.

Этот этап представляет собой важный шаг в процессе разработки, поскольку выбор конкретного метода создания модели определяет ее структуру и способ взаимодействия с базой данных.

После выбора "Конструктора EF из базы данных" открывается окно, где необходимо создать соединение с Microsoft SQL Server. На этом этапе мы активируем свойства подключения, внимательно настраивая параметры для успешного взаимодействия с базой данных.

1. Вводим имя сервера, на котором размещена база данных, в соответствующем поле.
2. Из выпадающего списка выбираем нужную базу данных, с которой мы будем работать в рамках проекта "NETKOplata".
3. В свойствах подключения, после ввода имени сервера и выбора базы данных, проводим проверку подключения с использованием соответствующей кнопки.

Этот этап предусматривает тщательную проверку корректности настроек подключения, что является критическим шагом для обеспечения успешного взаимодействия с базой данных.

После убеждения в корректности настроек подключения, мы завершаем процесс, нажимая на кнопку "Ок". Этот шаг закрывает окно настройки подключения и завершает процесс создания соединения с Microsoft SQL Server.

После успешного создания соединения, мастер моделей автоматически компилирует строку подключения к базе данных. На этом этапе нам остается выполнить последний шаг - указать название для Entity, который будет представлять собой модель данных, отображенную на базе данных.

В этом окне мастера мы внимательно выбираем подходящее название для Entity, ясно отражающее суть данных, которые оно будет представлять. Это имя будет использоваться в коде проекта и обеспечит понятность и структурированность в дальнейшей работе с данными.

После выбора названия для Entity, завершаем процесс, подтверждая наши выборы. Этот последний шаг фиксирует создание модели EDM, готовой к использованию в рамках информационной системы "NETKOplata".

После завершения предыдущих шагов, мы переходим к следующему этапу, где необходимо выбрать версию Entity Framework. В данной информационной системе было принято решение использовать 6 версию.

После выбора версии EF, открывается окно, где предстоит определить, какие конкретные объекты (таблицы или сущности) должны быть включены в модель EDM. В этом окне мы внимательно выбираем нужные таблицы, которые будут отражены в нашей модели данных.

После выбора объектов для включения в модель, вводим название для пространства имен модели. Это название обеспечит уникальность и легкость идентификации нашей модели в рамках проекта.

После завершения этих шагов, мы продвигаемся далее, подтверждая наши выборы. Этот этап заканчивается окончательным определением объектов, которые будут включены в нашу модель EDM, и уточнением пространства имен.

Совершив все необходимые действия, мы завершаем процесс, нажимая кнопку «Готово». После этого в структуре проекта появляется модель данных, представляющая собой отражение структуры базы данных. Также автоматически открывается диаграмма базы данных этой модели.

Этот момент является завершающим шагом в создании модели EDM для информационной системы "NETKOplata". В структуре проекта мы видим созданную модель, которая готова к использованию в рамках разработки приложения.

Открыв диаграмму базы данных, мы визуализируем структуру данных, ключевые сущности и связи между ними. Это инструмент, который облегчит понимание структуры базы данных и будет полезен при дальнейшей разработке и сопровождении проекта (рисунок 5).

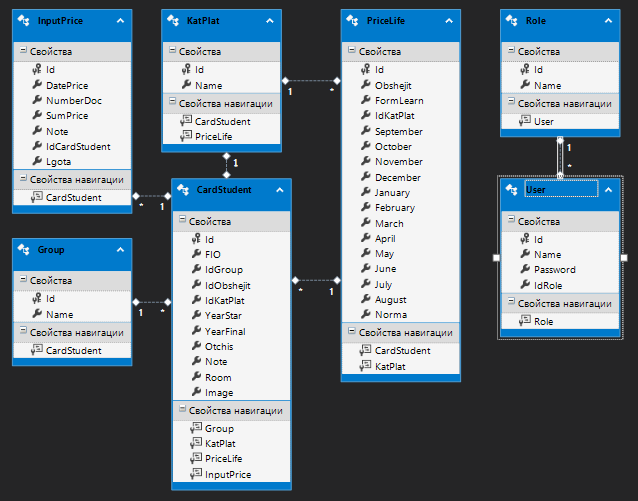


Рисунок 5 – Диаграмма модели

После успешного подключения к базе данных и создания модели для этой базы, инициализация класса "DbConnect" становится важным этапом для эффективного соединения модели с программой.

Для внедрения механизма авторизации в информационной системе было решено использовать две таблицы в базе данных: "User" и "Role". Эти таблицы содержат информацию о пользователях, присутствующих в системе.

Таблица "User" включает следующие поля:

* Id: уникальный идентификатор пользователя;
* Name: имя пользователя, также используемое в качестве логина;
* Password: пароль пользователя;
* IdRole: идентификатор роли пользователя;

Таблица "Role" содержит следующие поля:

* Id: уникальный идентификатор роли;
* Name: название роли;

Эти данные о пользователях и ролях будут использоваться для реализации системы авторизации в рамках проекта "NETKOplata". Сущности "User" и "Role" будут включены в модель данных, обеспечивая доступ к соответствующей информации в базе данных (рисунок 6).

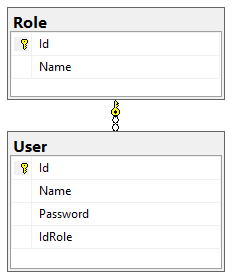


Рисунок 6 – Таблицы Role и User

На текущем этапе проекта была внедрена страница авторизации под названием "Authorization". Для этой страницы была разработана XAML-разметка, включающая следующие элементы управления:

* PasswordBox: этот элемент предназначен для ввода пароля. Его видимость была скрыта, чтобы обеспечить безопасность вводимых пользователем данных;
* TextBox: используется для ввода логина пользователем. Этот элемент предоставляет удобный интерфейс для ввода учетных данных;
* Label: выполняет роль обычной метки текста, предоставляя пользователю информацию и контекст относительно вводимых данных;
* Button: необходим для инициирования процесса авторизации. При его нажатии будут считаны введенные пользователем данные, и соответствующие события будут обработаны;

Эти элементы в совокупности обеспечивают пользовательское взаимодействие с системой авторизации. Пользователь может вводить свой логин и пароль, после чего, нажав на кнопку "Войти", система будет осуществлять процедуру авторизации (приложение А).

Создав разметку страницы авторизации, мы успешно завершили процесс формирования интерфейса для ввода учетных данных пользователей. Теперь пользователи могут воспользоваться этим интерфейсом для ввода своего логина и пароля, предоставляя необходимую информацию для процедуры авторизации (рисунок 7).

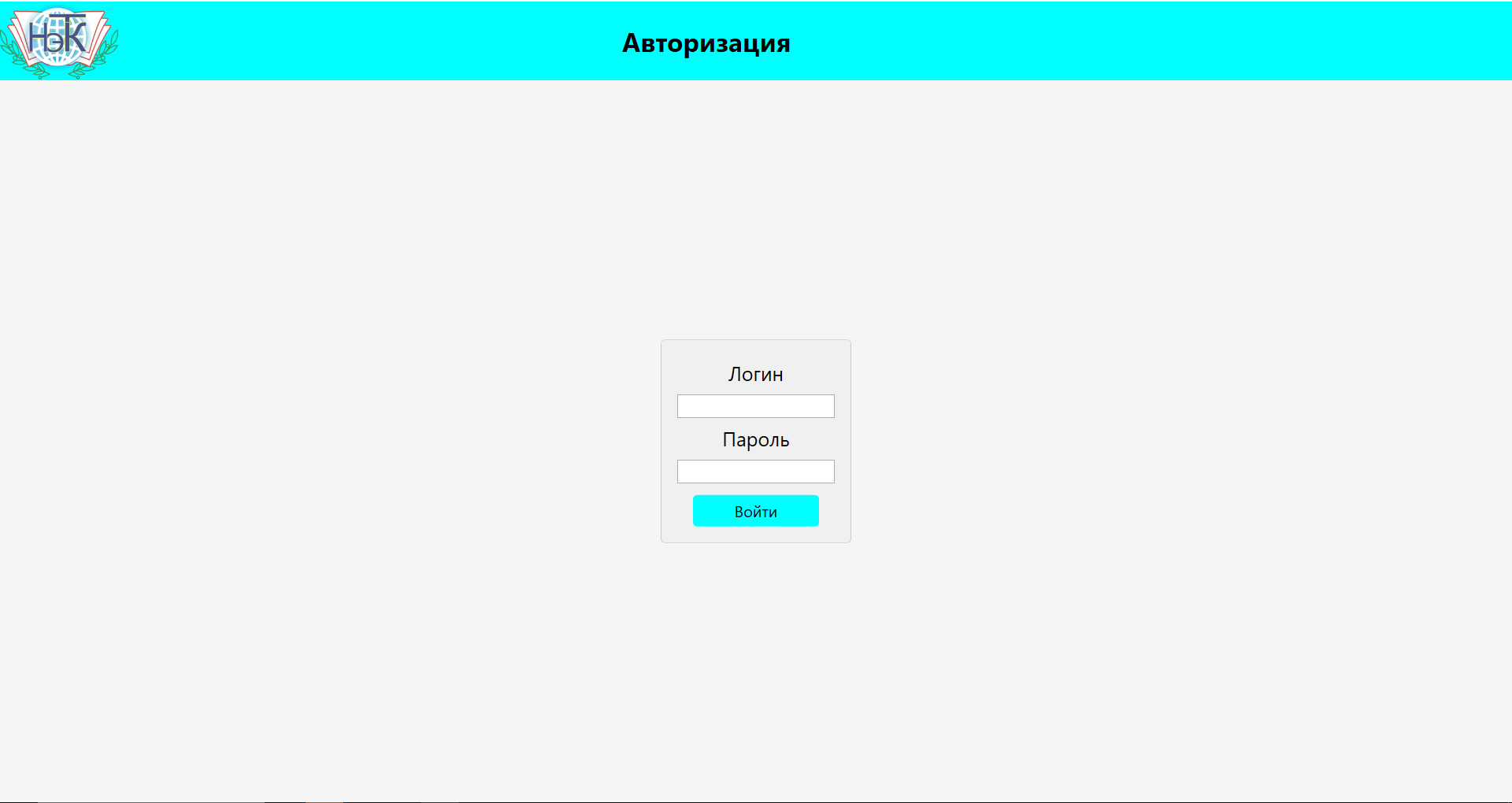


Рисунок 7 – Страница авторизации

В этом этапе мы реализуем код, который будет ответственен за обработку введенных пользователем данных при нажатии на кнопку "Войти". При вводе некорректных данных пользователю будет выдано уведомление о том, что такой пользователь не найден в системе. В случае успешной авторизации, пользователь получит приветственное сообщение и будет перенаправлен на следующую страницу (приложение Б).

В результате выполненной работы создан механизм авторизации, который гарантирует безопасный доступ к приложению и динамически отображает уведомления в зависимости от введенных данных.

Используя единое окно MainWindow, реализован пользовательский интерфейс, включающий объект Frame, предназначенный для навигации между страницами. Страницы интерфейса динамически загружаются во фрейм по мере необходимости или при взаимодействии с элементами, размещенными на страницах.

При запуске приложения автоматически открывается страница авторизации, на которой пользователю предоставляется возможность ввести учетные данные. После ввода система отображает уведомление о успешной авторизации. В случае, если предоставленные данные неверны, пользователь получает уведомление о том, что указанный аккаунт не существует (рисунок 8).

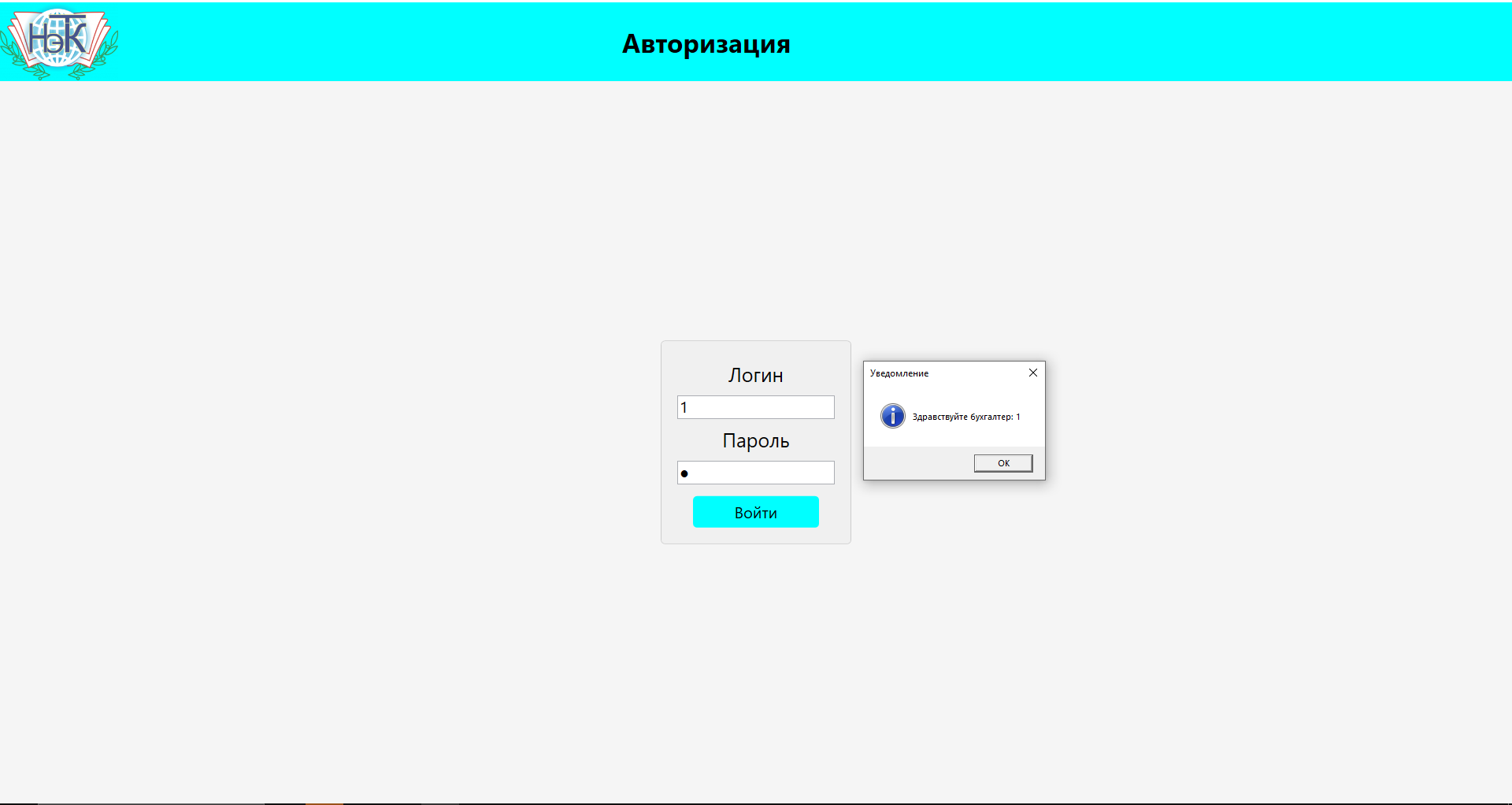


Рисунок 8 – Страница авторизации

После успешной авторизации пользователя происходит автоматический переход на главную страницу со списком всех студентов. На этой странице пользователю предоставляется удобный и интуитивно понятный интерфейс, предоставляющий доступ ко всем функциональным разделам приложения. Размещенные на странице кнопки "Группы", "Кат плательщика", "Оплата общежития" и " Оплата обучения " позволяют легко навигироваться между различными разделами системы, обеспечивая пользователю полный контроль над функциональностью приложения.

На странице представлена информация в виде таблицы, содержащей данные всех студентов. На этой странице реализованы разнообразные функциональности для удобного управления данными:

* Поиск студента:

Реализован механизм поиска, позволяющий быстро находить конкретного студента по различным критериям.

* Сортировка:

Предоставлена возможность сортировки данных по различным параметрам, что облегчает навигацию и анализ информации.

* Фильтрация:

Реализованы фильтры для точного выделения групп студентов или данных в соответствии с предварительно установленными критериями.

Таким образом, страница предоставляет не только обзор общей информации о студентах, но и обеспечивает широкие функциональности для управления данными, делая процесс администрирования более гибким и эффективным (рисунок 9).

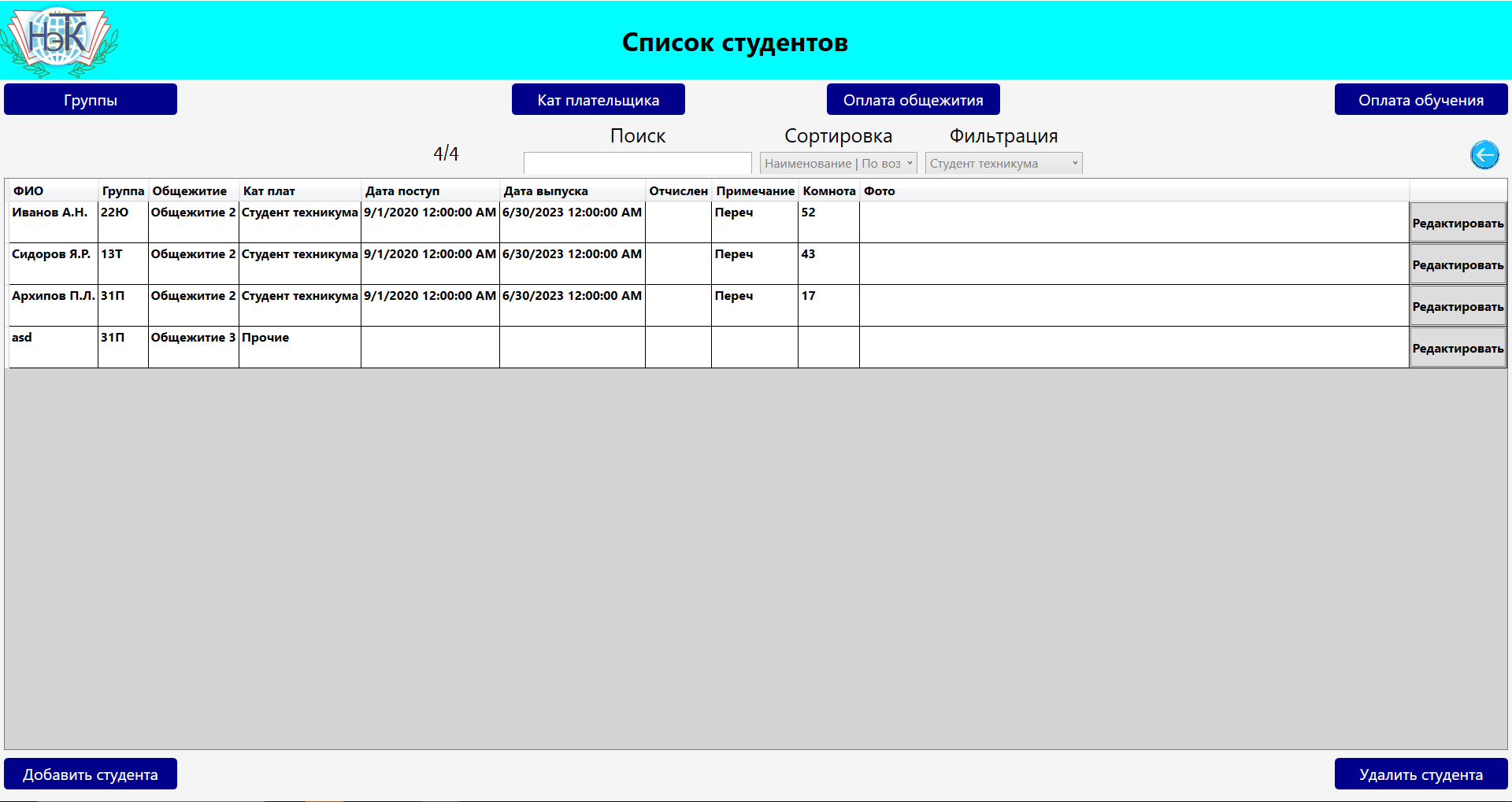


Рисунок 9 – Страница список студентов

Добавление нового студента.

На странице присутствует кнопка "Добавить", позволяющая бухгалтеру легко вводить данные новых студентов в систему (рисунок 10).

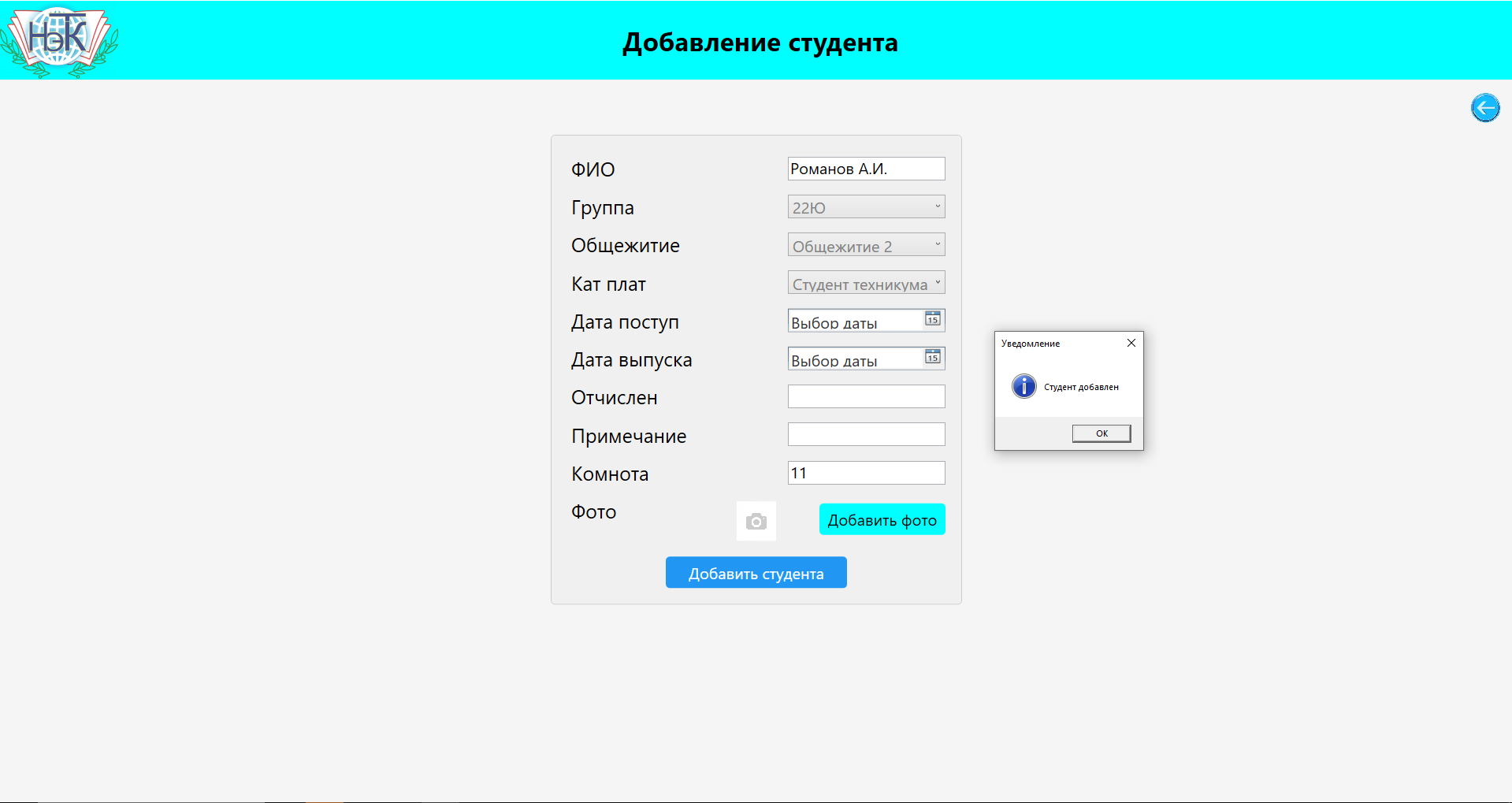


Рисунок 10 – Страница добавления

Удаление студента.

Для удаления выбранного студента предусмотрена кнопка "Удалить", обеспечивая бухгалтеру полный контроль над данными (рисунок 11).

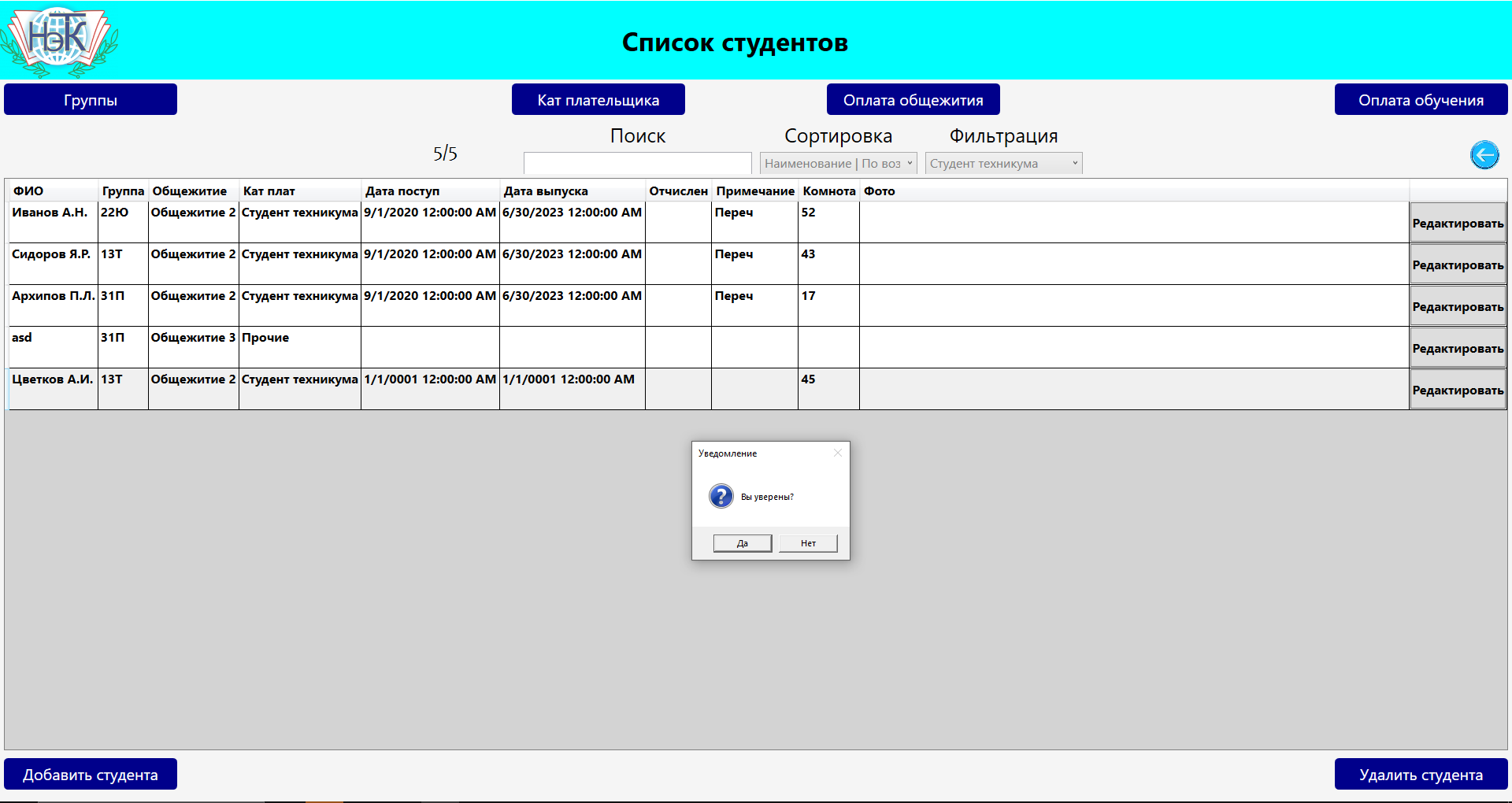


Рисунок 11 – Удаление студента

Редактирование данных.

При выборе определенного студента бухгалтер может воспользоваться функционалом кнопки "Редактировать", что обеспечивает возможность изменения информации о выбранном студенте (рисунок 12).

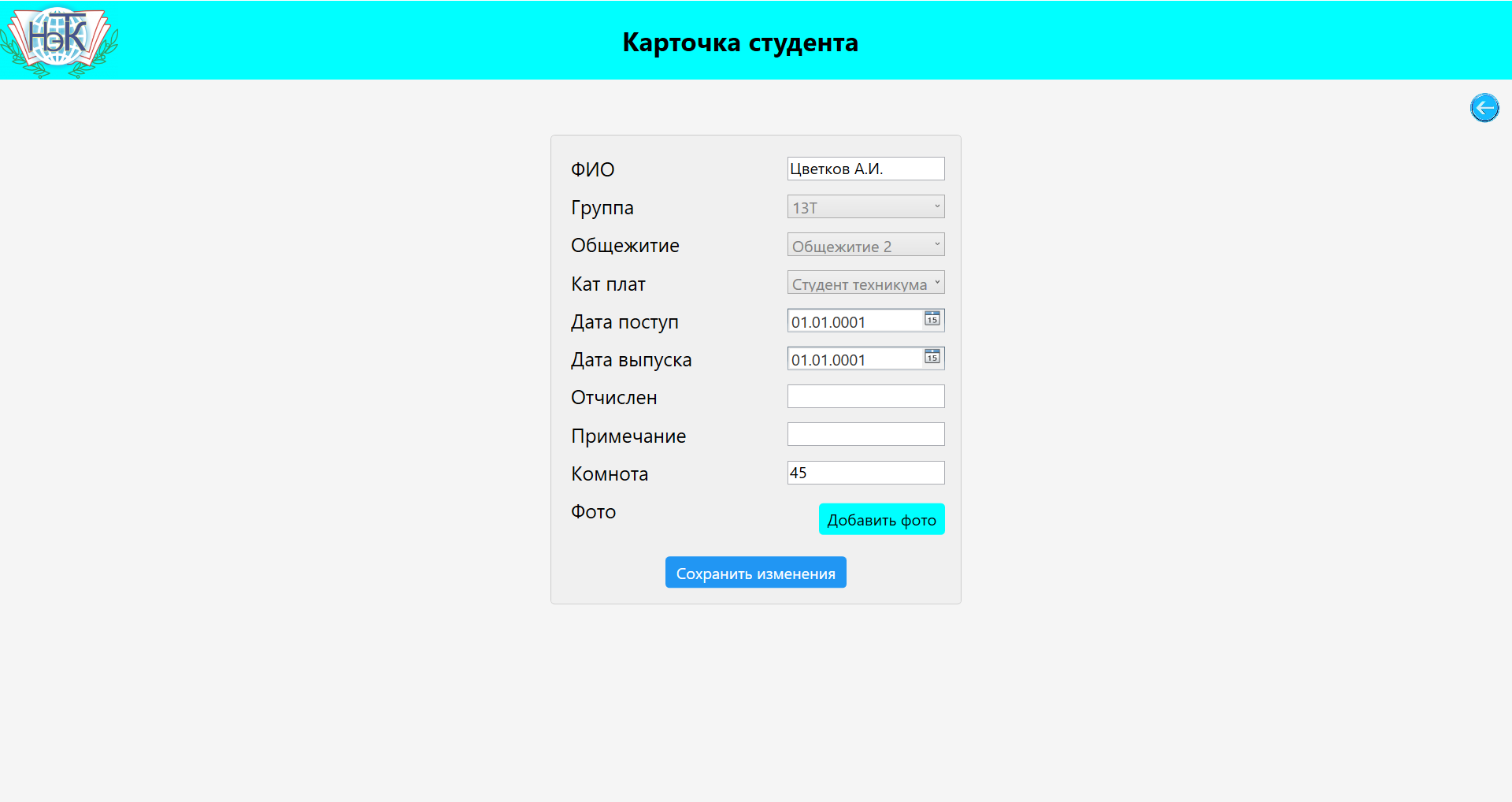


Рисунок 12 – Страница редактирования

Для остальных страниц "Группы", "Кат плательщиков" и "Оплата общежития" также реализован аналогичный интуитивно понятный и удобный интерфейс. Каждая страница предоставляет пользователю доступ к соответствующим разделам программы "NETKOplata", представленным в виде таблиц с данными и функционалом для управления информацией.

* Группы:

На странице "Группы" администратор может просматривать информацию о всех группах, выполнять сортировку и фильтрацию данных;

Предоставлены возможности добавления новой группы, удаления выбранной группы и редактирования данных о существующей группы;

* Кат плательщиков:

Страница "Кат плательщиков" предоставляет обзор всех категорий плательщиков в системе, с возможностью сортировки и фильтрации данных;

Реализованы функции добавления новой категории, удаления выбранной категории и редактирования информации о существующей категории;

* Оплата общежития:

На странице "Оплата общежития" бухгалтер может просматривать информацию об оплате, выполнять поиск, сортировку и фильтрацию данных;

Предоставлены возможности добавления новой категории оплаты, удаления выбранной категории и редактирования данных о существующей категории;

Разработанный интерфейс информационной системы "NETKOplata" обеспечивает пользователя удобным и эффективным способом взаимодействия с данными. Механизмы поиска, сортировки, фильтрации и управления записями позволяют оперативно осуществлять необходимые действия. Кроме того, интеграция функций добавления, удаления и редактирования данных делает систему гибкой и адаптивной к потребностям пользователей. Общий дизайн интерфейса содействует интуитивному восприятию, обеспечивая комфортное взаимодействие с информационной системой на всех уровнях её использования.

1. ТЕСТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ

Следующим шагом после разработки информационной системы является тестирование. Для тестирования необходимо составить 5 тест-кейсов, с помощью которых можно будет протестировать работоспособность разработанной программы.

Первый тест-кейс – проверка авторизации с правильным логином и паролем (Таблица 1):

Таблица 1 – Тест-кейс проверки авторизации с правильным логином и паролем

|  |  |
| --- | --- |
| Дата теста | 11.12.2023 |
| Приоритет теста | Высокий |
| Название тестирования/Имя | Проверка авторизации с правильным логином и паролем. |
| Этапы тестирования | 1. Запустить приложение; 2. Ввести правильный логин (1); 3. Ввести правильный пароль (1); 4. Нажать на кнопку «Войти»; 5. Получить сообщение «Здравствуйте, бухгалтер 1»; 6. Переход на главную страницу бухгалтера. |
| Тестовые данные | Логин –1;  Пароль -1. |
| Ожидаемый результат | Вход в систему с получаемым сообщением «Здравствуйте, бухгалтер 1». |
| Фактический результат | Произошел успешный вход в систему. Уведомление «Здравствуйте, бухгалтер 1» получено. |

Второй тест-кейс – проверка авторизации с неправильным логином и паролем (Таблица 2):

Таблица №2 – Тест-кейс проверки авторизации с неправильным логином и паролем

|  |  |
| --- | --- |
| Дата теста | 11.12.2023 |
| Приоритет теста | Высокий |
| Название тестирования/Имя | Проверка авторизации с неправильным логином и паролем. |
| Этапы тестирования | 1. Запустить приложение; 2. Ввести логин (asd); 3. Ввести пароль (asd); 4. Нажать кнопку «Войти»; 5. Получить сообщение «Такой пользователь не найден»; |
| Тестовые данные | Логин – asd;  Пароль –asd; |
| Ожидаемый результат | Вывод сообщения «Такой пользователь не найден», не допуск к последующему использованию приложения. |
| Фактический результат | Произошел успешный вывод сообщения «Такой пользователь не найден» с последующим не допуском к работе приложения. |

Третий тест-кейс – проверка добавления студента (Таблица 3):

Таблица 3 – Тест-кейс проверки добавления студента

|  |  |
| --- | --- |
| Дата теста | 11.11.2023 |
| Приоритет теста | Высокий |
| Название тестирования/Имя | Проверка добавления пользователя. |
| Этапы тестирования | 1. Запустить приложение; 2. Ввести логин (1); 3. Ввести пароль (1); 4. Нажать на кнопку «Добавить судента»; 5. Ввести ФИО (qwe); 6. Выбрать группу (22Ю); 7. Выбрать общежитие (Общежитие 2); 8. Выбрать кат плат (Студент техникума); 9. Ввести дату поступления (01.09.2021); 10. Ввести дату выпуска (30.06.2024); 11. Ввести или не вводить отчислен (); 12. Ввести или не вводить примечание (); 13. Ввести или не вводить комната (); 14. Выбрать или не выбирать фотографию (); 15. Нажать кнопку «Добавить студента». |
| Тестовые данные | Логин – 1;  Пароль –1;  ФИО – qwe;  Группа – 22Ю;  Общежитие – Общежитие 2;  Кат плат – Студент техникума;  Дата поступления – 01.09.2021;  Дата выпуска – 30.06.2024;  Отчислен – 30.06.2024;  Примечание – 30.06.2024;  Комната – 30.06.2024;  Фотография – 30.06.2024; |
| Ожидаемый результат | Студент с введёнными данными будет добавлен в список студентов. Вывод сообщения об успешном добавлении студента. |
| Фактический результат | Добавление студента произошло успешно. Сообщение об успешном добавлении студента выведено. |

Четвёртый тест-кейс – проверка удаления студента из списка (Таблица 4):

Таблица 4 – Тест-кейс проверки удаления документа из списка

|  |  |
| --- | --- |
| Дата теста | 11.12.2023 |
| Приоритет теста | Высокий |
| Название тестирования/Имя | Проверка удаления студента из списка. |
| Этапы тестирования | 1. Запустить приложение; 2. Ввести правильный логин (1); 3. Ввести правильный пароль (1); 4. Нажать на кнопку «Войти»; 5. Выбрать студента в таблице; 6. Нажать кнопку «Удалить». |
| Тестовые данные | Логин – 1;  Пароль - 1; |
| Ожидаемый результат | Выбранный студент будет удалён из списка студентов. Вывод сообщения об успешном удалении студента. |
| Фактический результат | Произошло успешное удаление студента. Уведомление об успешном удалении студента получено. |

Пятый тест-кейс – проверка редактирования студента из списка (Таблица 5):

Таблица 5 – Тест-кейс проверки редактирования студента из списка

|  |  |
| --- | --- |
| Дата теста | 11.12.2023 |
| Приоритет теста | Высокий |
| Название тестирования/Имя | Проверка редактирования студента из списка. |
| Этапы тестирования | 1. Запустить приложение; 2. Ввести правильный логин (1); 3. Ввести правильный пароль (1); 4. Нажать на кнопку «Войти»; 5. Нажать на кнопку «Редактировать»; 6. Редактировать группу студента (13Т); 7. Нажать кнопку «Сохранить изменения» |
| Тестовые данные | Логин – 1;  Пароль – 1;  Группа студента – 13Т; |
| Ожидаемый результат | Группа выбранного студента будет изменена. Вывод сообщения об успешном изменении студента. |
| Фактический результат | Произошло успешное изменение группы студента. Уведомление об успешном изменении студента получено. |

Таким образом, были проверены следующие функции приложения:

- авторизация в системе с правильным логином и паролем;

- авторизация в системе с неправильным логином и паролем;

- добавление студента;

- удаление студента;

- редактирование студента;

Во всех пяти тест-кейсах ожидаемый результат совпадает с фактическим результатом.

ВЫВОДЫ

В ходе выполнения нашего проекта были успешно достигнуты основные цели - разработки информационной системы "NETKOplata" для эффективного учета, контроля и планирования работы с данными в колледже. Также система направлена на повышение безопасности данных и обеспечение оперативного доступа к информации.

Для достижения поставленных задач мы выполнили ряд этапов, включая моделирование структуры данных, выбор методов и средств разработки, подключение к базе данных, создание ролей и авторизации, а также разработку пользовательских интерфейсов. Затем мы провели тестирование системы.

Реализация системы "NETKOplata" обеспечивает безопасность данных и контроль доступа через подключение к базе данных и установку ролей с соответствующими правами. Пользовательские интерфейсы для разных категорий пользователей, способствуют удобству использования системы.

Проведенное тестирование подтвердило стабильность и надежность системы "NETKOplata". Выявленные ошибки были успешно устранены, и система полностью соответствует требованиям и целям проекта. Тестирование также выявило возможные улучшения, которые могут быть внедрены в будущем для улучшения производительности и функциональности.

В итоге, разработанная информационная система "NETKOplata" успешно решает поставленные задачи, автоматизируя учет, контроль и планирование работы с данными в колледже. Она не только повышает безопасность данных, но и обеспечивает быстрый доступ к необходимой информации. С использованием современных методов и средств разработки, интуитивно понятного пользовательского интерфейса и успешного тестирования, система "NETKOplata" представляет собой надежный и эффективный инструмент для отдела кадров колледжа.

ПРИЛОЖЕНИЕ А

КОД РАЗМЕТКИ СТРАНИЦЫ «Authorization»

<Page x:Class="NETKOplata.Pages.Authoriz"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:NETKOplata.Pages"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="Авторизация">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="\*" />

<RowDefinition Height="Auto" />

<RowDefinition Height="\*" />

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*" />

<ColumnDefinition Width="Auto" />

<ColumnDefinition Width="\*" />

</Grid.ColumnDefinitions>

<Border Grid.Row="1" Grid.Column="1" Background="#F0F0F0" CornerRadius="5" BorderBrush="#CCCCCC" BorderThickness="1">

<Grid Margin="20">

<Grid.RowDefinitions>

<RowDefinition Height="Auto" />

<RowDefinition Height="Auto" />

<RowDefinition Height="Auto" />

<RowDefinition Height="Auto" />

<RowDefinition Height="Auto" />

</Grid.RowDefinitions>

<Label Content="Логин" Grid.Row="0" HorizontalAlignment="Center" FontSize="25" Margin="0,0,0,5" />

<TextBox Name="TxbLogin" Grid.Row="1" HorizontalAlignment="Center" Width="200" Height="30" FontSize="20" Margin="0,0,0,5"/>

<Label Content="Пароль" Grid.Row="2" HorizontalAlignment="Center" FontSize="25" Margin="0,0,0,5"/>

<PasswordBox Name="PsbPassword" Grid.Row="3" HorizontalAlignment="Center" Width="200" Height="30" FontSize="20" Margin="0,0,0,5" />

<Button Name="BtnLogin" Click="BtnLogin\_Click" Grid.Row="4" HorizontalAlignment="Center" Width="160"

Height="40" Margin="0,10,0,0" Background="#00FFFF" Foreground="Black" Content="Войти" FontSize="20">

<Button.Template>

<ControlTemplate TargetType="Button">

<Border Background="{TemplateBinding Background}" CornerRadius="5" Padding="5">

<ContentPresenter HorizontalAlignment="Center" VerticalAlignment="Center" />

</Border>

</ControlTemplate>

</Button.Template>

</Button>

</Grid>

</Border>

</Grid>

</Page>

ПРИЛОЖЕНИЕ Б

КОД СТРАНИЦЫ «Authorization»

namespace NETKOplata.Pages

{

/// <summary>

/// Логика взаимодействия для Authoriz.xaml

/// </summary>

public partial class Authoriz : Page

{

public Authoriz()

{

InitializeComponent();

}

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

{

try

{

var userObj = DbConnect.entObj.User.FirstOrDefault(x => x.Name == TxbLogin.Text && x.Password == PsbPassword.Password);

if (userObj == null)

{

MessageBox.Show("Такой пользователь не найден.",

"Уведомление",

MessageBoxButton.OK,

MessageBoxImage.Information);

}

else

{

switch (userObj.IdRole)

{

case 1:

MessageBox.Show("Здравствуйте бухгалтер: " + userObj.Name,

"Уведомление",

MessageBoxButton.OK,

MessageBoxImage.Information);

FrameApp.frmObj.Navigate(new Pages.ListStudents());

break;

case 2:

MessageBox.Show("Здравствуйте классный руководитель: " + userObj.Name,

"Уведомление",

MessageBoxButton.OK,

MessageBoxImage.Information);

FrameApp.frmObj.Navigate(new Pages.ListStudents());

break;

}

}

}

catch (Exception ex)

{

MessageBox.Show("Ошибка работы приложения: " + ex.Message.ToString(),

"Критический сбой работы приложения",

MessageBoxButton.OK,

MessageBoxImage.Warning);

throw;

}

}

}

}