МИНИСТЕРСТВО ОБРАЗОВАНИЯ КИРОВСКОЙ ОБЛАСТИ

Кировское областное государственное профессиональное образовательное бюджетное учреждение

«Слободской колледж педагогики и социальных отношений»

**КУРСОВОЙ ПРОЕКТ**

по ПМ 01 «Разработка программных модулей» на тему:

**РАЗРАБОТКА ПРОГРАММНОГО МОДУЛЯ ДЛЯ УЧЕТА**

**ПРОПУСКОВ УЧЕБНЫХ ЗАНЯТИЙ**

Выполнила: Кротова Ксения Николаевна

Специальность 09.02.07

Информационные системы и программирование

Группа 21П-1

Форма обучения: очная

Руководитель: Пентин Николай Сергеевич

Дата защиты курсового проекта:

Председатель ПЦК:

Оценка за защиту курсового проекта:

Слободской

2024

ОГЛАВЛЕНИЕ

[ВВЕДЕНИЕ 3](#_Toc185574418)

[АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 5](#_Toc185574419)

[РАЗРАБОТКА ТЕХНИЧЕСКОГО ЗАДАНИЯ 9](#_Toc185574420)

[ОПИСАНИЕ АЛГОРИТМОВ И ФУНКЦИОНИРОВАНИЯ ПРОГРАММЫ 13](#_Toc185574421)

[ТЕСТИРОВАНИЕ ПРОГРАММНОГО МОДУЛЯ 19](#_Toc185574422)

[РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ 22](#_Toc185574423)

[ЗАКЛЮЧЕНИЕ 26](#_Toc185574424)

[СПИСОК ЛИТЕРАТУРЫ 27](#_Toc185574425)

[ПРИЛОЖЕНИЯ 28](#_Toc185574426)

[Приложение 1 29](#_Toc185574427)

[Приложение 2 37](#_Toc185574428)

ВВЕДЕНИЕ

В современном образовательном процессе одной из ключевых задач является обеспечение академической успеваемости студентов, а также эффективный контроль за их посещаемостью. Учебные заведения, будь то школы, колледжи или университеты, сталкиваются с необходимостью систематически фиксировать и отслеживать посещаемость занятий. Проблема учета посещаемости учебных занятий актуальна как для студентов, так и для педагогов и администрации учебных заведений, поскольку стабильное посещение занятий влияет на качество образовательного процесса и общую успеваемость учащихся.

Важность учета посещаемости студентов в учебном процессе не может быть недооценена. Пропуски занятий напрямую влияют на уровень знаний студентов и их успеваемость, а также могут служить индикатором ряда проблем, включая отсутствие мотивации, личные трудности или даже проблемы со здоровьем. Получая своевременную информацию о пропусках, преподаватели и администрация могут принимать меры, направленные на улучшение ситуации.

Исторически ведение учета посещаемости занимало значительное количество ресурсов, включая как человеческие, так и временные. Традиционные методы, такие как журнал или тетрадь для записи посещаемости, имеют свои ограничения: они подвержены ошибкам, могут быть неправильно заполнены и не обеспечивают необходимого уровня удобства и эффективности. Век цифровых технологий диктует новые требования к процессам, связанным с администрированием образовательных учреждений. На сегодняшний день многие учебные заведения переходят на автоматизированные системы учета, что позволяет значительно упростить и улучшить ведение документации.

Таким образом, программный модуль для учета посещаемости учебных занятий может стать инструментом, который поможет в реализации проактивного подхода к поддержке студентов, позволяя выявлять значимые закономерности и тенденции.

Цель курсового проекта – разработка программного модуля для учета посещаемости учебных занятий студентами.

Задачи исследования:

* Описать предметную область.
* Разработать технического задание на создание программного продукта.
* Описать архитектуру программы.
* Описать алгоритмы и функционирование программы.
* Провести тестирование и опытную эксплуатацию.
* Разработать руководство оператора

Объект исследования – процесс учета и мониторинга посещаемости студентов в образовательной среде.

Предмет исследования – разработка программной системы для учета и мониторинга посещаемости учащихся.

Методы исследования: системный анализ и функциональное моделирование.

Информационную систему исследования составили официальные нормативно-правовые источники, данные об использовании современных информационных систем. Структура работы состоит из введения, трех глав, заключения, списка используемой литературы и приложений.

АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

Преподаватель – работник высших, средних специальных и профессионально-технических учебных заведений, ведущий какой-либо предмет и воспитательную работу (в общеобразовательной школе – учитель).

Студент – учащийся высшего, в некоторых государствах и странах и среднего учебного заведения.

Администратор - специалист, отвечающий за управление и обслуживание клиентского приложения и базы данных.

В процессе анализа предметной области было проведено исследование, в ходе которого была разработана диаграмма вариантов использования, отражающая пользователей разрабатываемого программного модуля(рисунок 1).

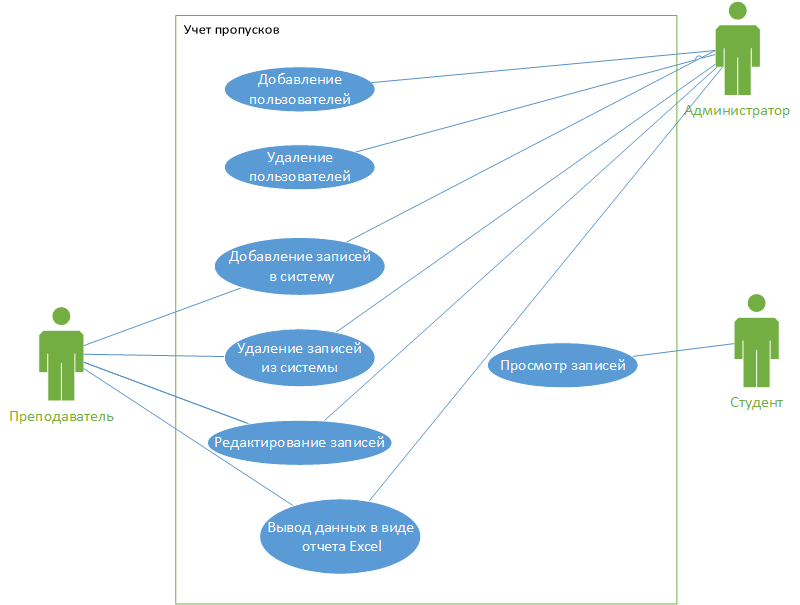
****

Рисунок 1 - Диаграмма вариантов использования

**Обзор аналогов**

Учебный учет – программа для учета посещаемости и успеваемости в помощь классным руководителям и заведующим отделения. Программа позволит подготовить электронные журналы, сформировать отчеты посещаемости занятий и текущей успеваемости.

Плюсами данного программного продукта является ввод из журнала оценок и пропусков студентов (учеников), составление ведомостей, автоматическое составление отчетов по успеваемости и посещаемости, а также наглядное представление качественной успеваемости по группам (классам) и специальностям в виде графиков. Есть возможность настройки создания шаблонов печатных форм в Word. Минусом данного программного продукта является ее старый интерфейс, а также высокая стоимость – за лицензию для одного компьютера вам придется заплатить около 8000 рублей, а за сетевую лицензию около 13000 рублей.

UUStud - программа предназначена для учета успеваемости студентов в высших учебных заведениях. Плюсами данного программного продукта является простой интерфейс, а также вывод на печать отчета. Минусом данного программного продукта является очень медленная работа.

После изучения аналогов разрабатываемого программного модуля были выделены следующие сущности и их атрибуты:

**User\_type:**

* Id\_types (int);
* Type\_Name (varchar);
* Description (varchar).

**User:**

* Id\_user (int);
* Login (varchar);
* Password (varchar);
* Id\_student (int);
* Type\_User (int).

**Student:**

* Id\_student (int);
* Surname (varchar);
* Name (varchar);
* Middle\_Name (varchar);
* Group (int).

**Group:**

* Id\_group (int);
* Group\_Name (varchar);
* Specialization (int).
* Specialization:
* Id\_specialization (int);
* Number\_Specialization (varchar);
* Specialization\_Name (varchar);
* Description (varchar).

**Attendance:**

* Id\_attendence (int);
* Id\_student (int);
* Subject (int);
* Data (datetime);
* Reason (varchar);
* Description (varchar).

**Subject:**

* id\_subject (int);
* Name (varchar).

На основании выделенных сущностей была создана модель базы данных в СУБД Microsoft SQL Server 2019 Express (Рисунок 2):

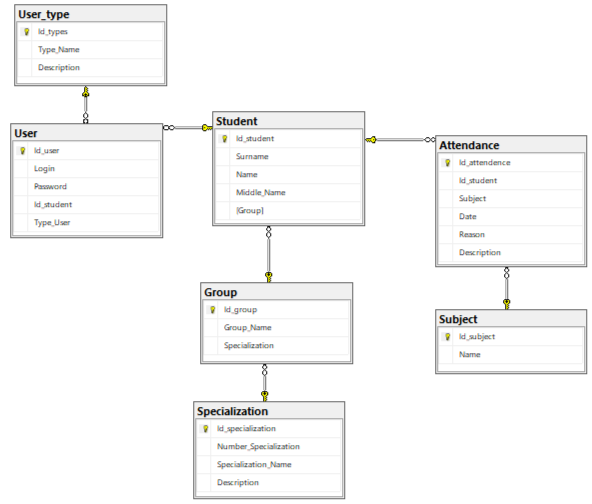


Рисунок 2 – Модель базы данных в системе

РАЗРАБОТКА ТЕХНИЧЕСКОГО ЗАДАНИЯ

Наименование программы – «Учет посещаемости». Программный модуль предназначен для ведения и автоматизации учёта посещаемости учебных занятий студентами [ГОСТ 19.201-78].

Разработка программы ведется на основании учебного плана и перечня тем утвержденных на заседании предметно цикловой комиссии информатики и программирования.

Функциональным назначением программы является ведение и автоматизация учёта посещаемости учебных занятий студентами.

Программа должна обеспечивать возможность выполнения перечисленных ниже функций:

* Просмотр данных о группах, студентах, предметах, специальностях и учете посещаемости.
* Ведение учёта посредством добавления информации о пропусках занятий обучающимися.
* Возможность удаления, редактирования данных.
* Возможность сохранения данных в виде отчета.

Надежное (устойчивое) функционирование программы должно быть обеспечено выполнение заказчиком совокупности организационно-технических мероприятий, перечень которых приведен ниже:

* организация бесперебойного питания технических средств;
* использование лицензионного программного обеспечения;
* отсутствие вредоносного программного обеспечения, наличие антивирусной программы;
* соблюдение правил и требований по эксплуатации технических средств.

Время восстановления после отказа, вызванного сбоем электропитания технических средств (иными внешними факторами), не фатальным сбоем (не крахом) операционной системы, не должно превышать 5 минут при условии соблюдения условий эксплуатации технических и программных средств.

Время восстановления после отказа, вызванного неисправностью технических средств, фатальным сбоем (крахом) операционной системы, не должно превышать времени, требуемого на устранение неисправностей технических средств и переустановки программных средств.

Отказы программы возможны вследствие некорректных действий оператора (пользователя) при взаимодействии с операционной системой. Во избежание возникновения отказов программы по указанной выше причине следует обеспечить работу пользователя без предоставления ему административных привилегий.

Климатические условия эксплуатации, при которых должны обеспечиваться заданные характеристики, должны удовлетворять требованиям, предъявляемым к техническим средствам в части условий их эксплуатации.

В состав технических средств должен входить IBM-совместимый персональный компьютер (ПЭВМ), включающий себя:

* процессор с тактовой частотой, 1 ГГц, не менее;
* оперативную память объемом 512 Мб, не менее;
* жесткий диск со свободным местом 500 Мб, не менее;
* монитор, с разрешением экрана 1024\*768, не менее;
* оптический привод;
* компьютерная мышь;
* клавиатура;

Исходные коды программы должны быть реализованы на языке С#. В качестве интегрированной среды разработки программы должна быть использована среда программирования Microsoft Visual Studio 2022, в качестве СУБД для разработки базы данных должен быть использован Microsoft SQL Server 2019 Express.

Системные программные средства, используемые программой, должны быть представлены лицензионной локализованной версией операционной системы Windows 7/8/10/11.

Программное обеспечение поставляется в виде изделия на CD диске.

Упаковка программного изделия должна осуществляться в упаковочную тару предприятия-изготовителя компакт диска

Требования к транспортировке и хранению должны соответствовать условиям эксплуатации носителей, на которых находится программный продукт.

Программа должна обеспечивать взаимодействие с пользователем посредством графического пользовательского интерфейса.

Предварительный состав программной документации включает в себя следующие документы:

* техническое задание;
* руководство оператора.

Разработка должна быть проведена в следующие стадии и этапы:

1. Анализ требований:

На стадии анализ требований формулируются цели и задачи проекта. Создается основа для дальнейшего проектирования

1. Проектирование:

На стадии проектирование должны быть выполнены перечисленные ниже этапы работ:

* разработка программной документации;

На этапе разработка программной документации должна быть выполнена разработка технического задания.

При разработке технического задания должны быть выполнены перечисленные работы: постановка задачи, определение и уточнение требований к техническим средствам, определение требований к программе, определение стадий, этапов и сроков разработки программы и документации на нее, выбор языков программирования.

* разработка алгоритма программы;

На этапе разработки алгоритма программы должен быть разработан алгоритм работы программы.

* кодирование;

На стадии кодирования происходит реализация алгоритмов в среде программирования.

* тестирование и отладка.

На стадии тестирование и отладка происходит проверка алгоритмов, реализованных в программе на работоспособность в различных ситуациях. Исправление выявленных ошибок, повторное тестирование.

Приемо-сдаточные испытания должны проводиться при использовании технических средств. Приемка программы заключается в проверке работоспособности программы путем ввода реальных или демонстрационных данных.

Во время приемки работы разработчик предоставляет программу и документацию, которая к ней прилагается. Проводятся испытания программы, при успешных испытаниях программа вводится в эксплуатацию. При ошибках, недопустимых для успешной работы программного продукта – отправляется на доработку.

Было описано техническое задание, содержащее в себе информацию о программном продукте, его функциях, эксплуатации и требования, которые должны учитываться при создании программы и документации к ней.

ОПИСАНИЕ АЛГОРИТМОВ И ФУНКЦИОНИРОВАНИЯ ПРОГРАММЫ

Все данные выводящиеся, добавляющиеся, удаляемые, редактируемые и используемые в программе, хранятся в базе данных SQL. Подключение к ней осуществляется посредством технологии ADO.NET. Модель базы данных хранится в программном модуле (рисунок 3).

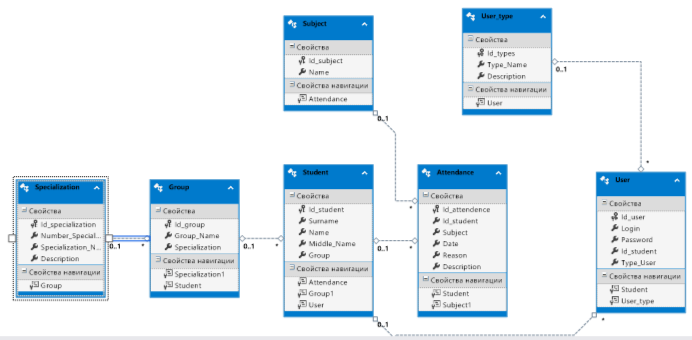


Рисунок 3 – Модель Entity Framework для привязанной к модулю базы данных SQL

При запуске программы происходит отображение формы авторизации, на которой пользователю предлагается ввести логин и пароль от существующей учётной записи (рисунок 4).

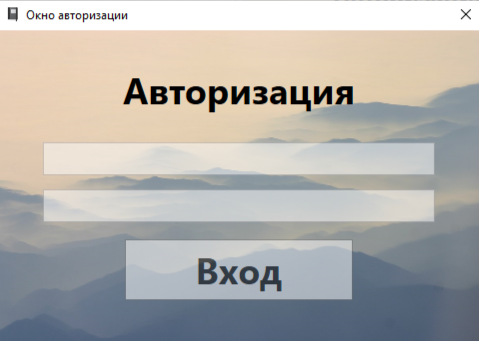


Рисунок 4 - Окно авторизации пользователя

В случае нажатия на кнопку «Вход» при корректном вводе данных, в зависимости от роли, назначенной пользователю в системе, откроется окно с соответствующими для пользователя возможностями.

Возможности пользователе в системе:

Администратор

* Добавление, удаление, изменение записей
* Вывод в Excel
* Добавление новых пользователей

Преподаватель

* Добавление, удаление, изменение и просмотр записей
* Вывод в Excel

Студент

* Просмотр записей

После авторизации для каждого типа пользователя открывается форма, имеющая свой функционал у всех ролей.

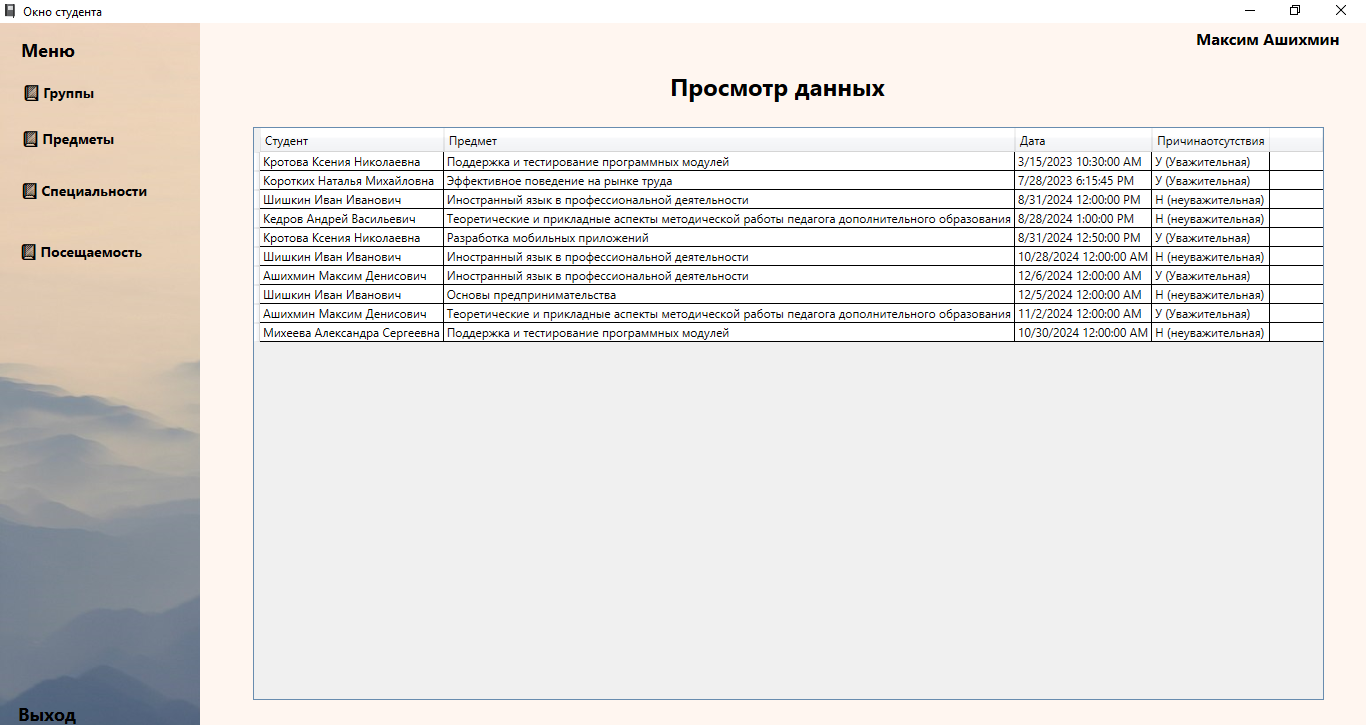


Рисунок 5 - Окно с функционалом студента

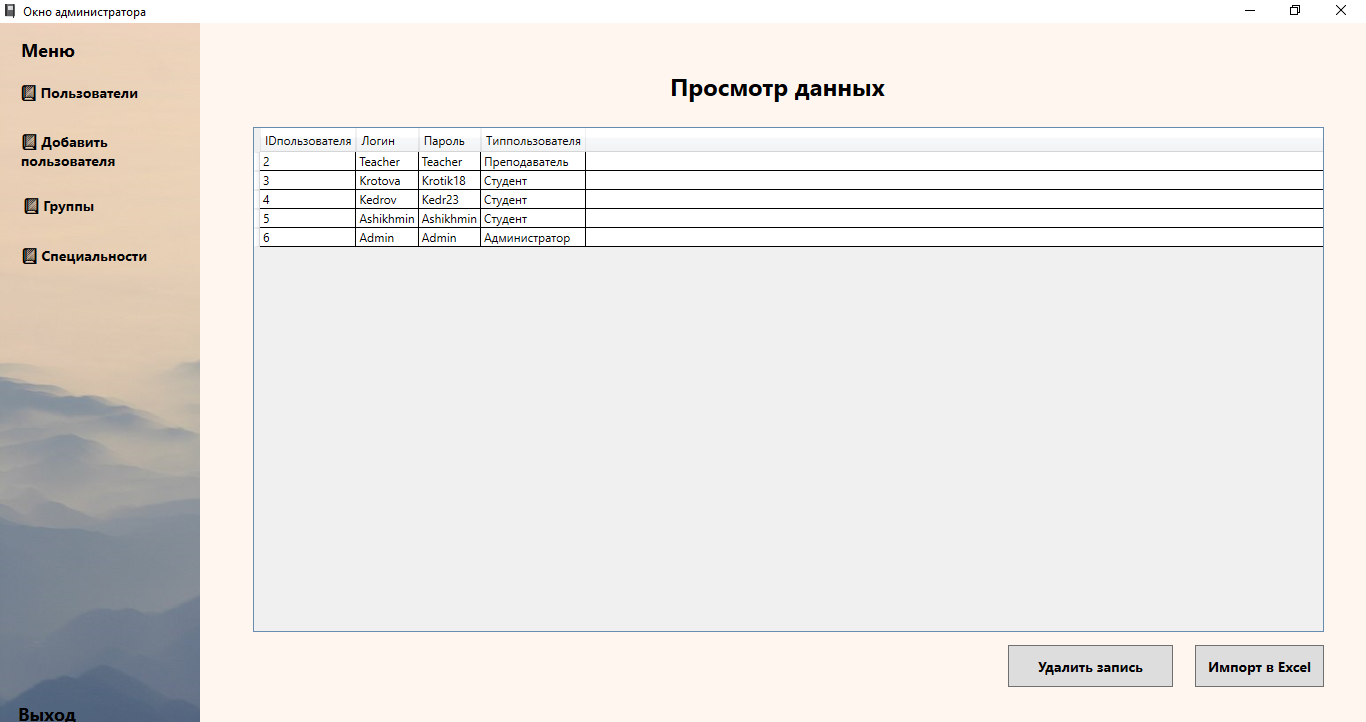


Рисунок 6 - Окно с функционалом администратора

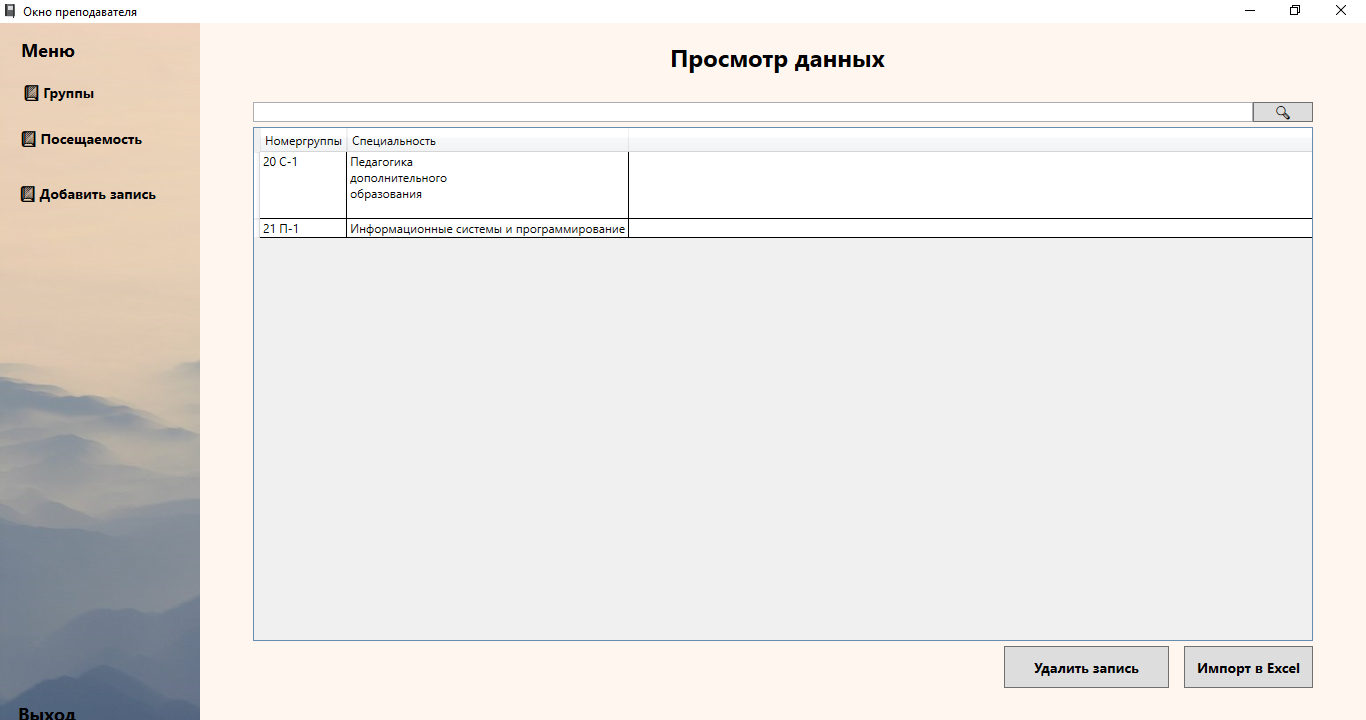


Рисунок 7 - Окно с функционалом преподавателя

Рассмотрим подробнее функционал преподавателя. Кнопка «Группы» запускает процесс показа списка групп, выгружая результаты на форму.

Кнопка «Посещаемость» запускает процесс показа записей о пропусках, выгружая результаты на форму (рисунок 8).

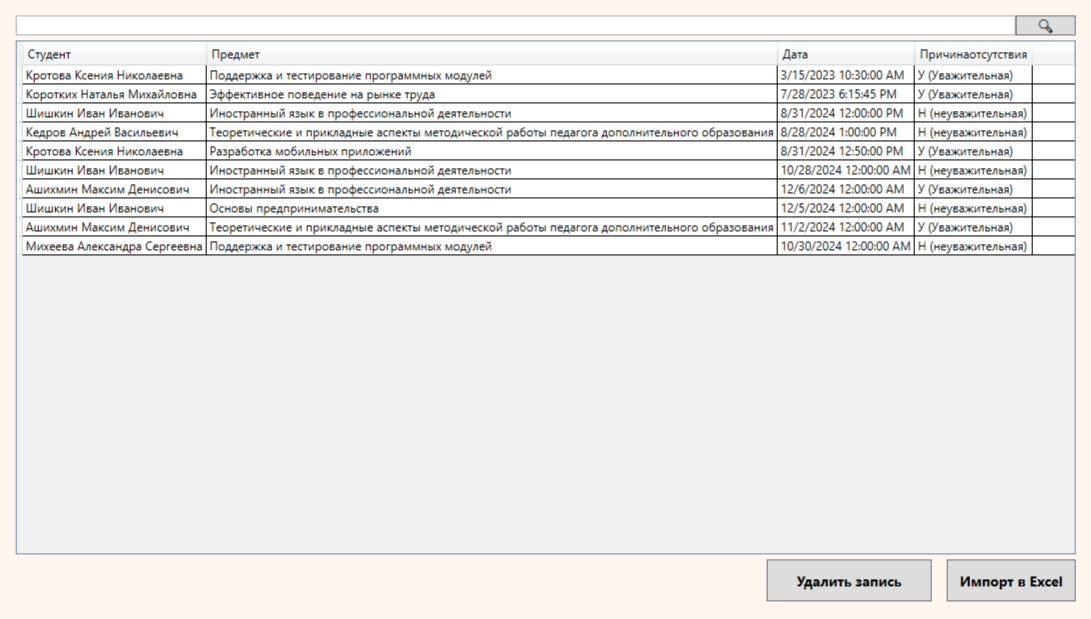


Рисунок 8 – Просмотр записей о посещаемости студентов

Кнопка «Импорт в Excel» запускает процесс сохранения данных о посещаемости студентов в файл xlsx (рисунок 9). Для работы с файлами Excel била использована библиотека EpPlus.

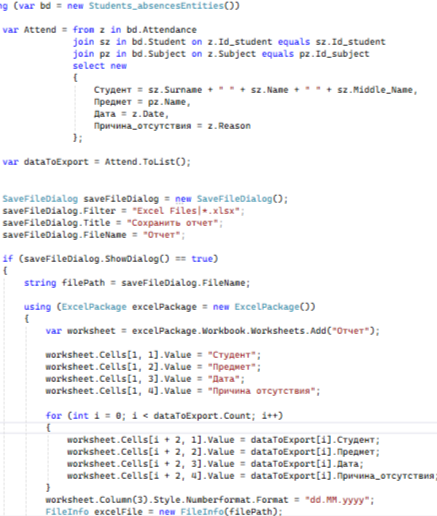


Рисунок 9 – Импорт записей в отчет

Также реализован функционал удаления записи (рисунок 10).

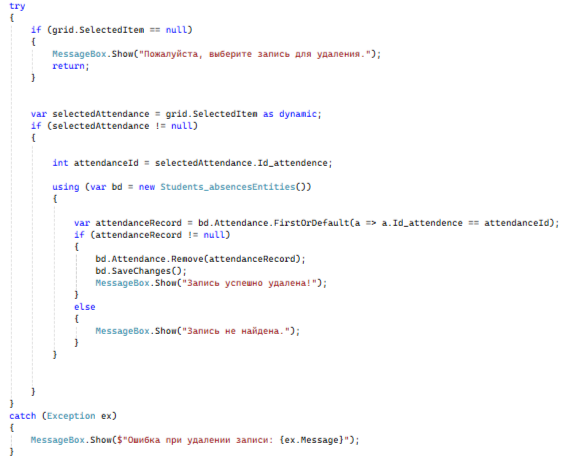


Рисунок 10 – Метод удаления записи

Кнопка «Добавить запись» запускает новок окно для ввода данных о посещаемости студентов (рисунок 11). Все данные сохраняются корректно.

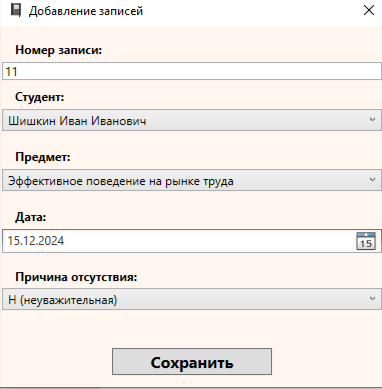


Рисунок 11 – Добавление записей о посещаемости

Также в программе реализован метод поиска по записям (рисунок 12).

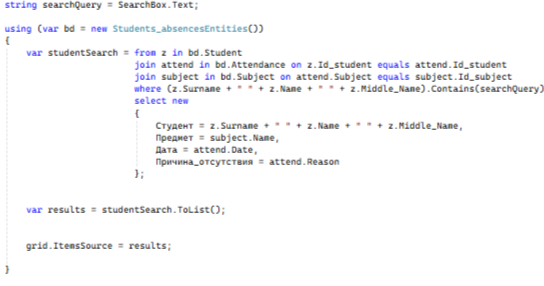


Рисунок 12 – Метод поиска

В этом параграфе были описаны основные методы и алгоритмы функционирования программы и библиотеки, используемые в программе.

ТЕСТИРОВАНИЕ ПРОГРАММНОГО МОДУЛЯ

Для проведения тестирования программы мною было произведено базовое тестирование во время разработки программы. При тестировании был выявлен ряд ошибок, которые возникли в ходе выполнения программы.

* Попытка удаления записи о посещаемости студентов

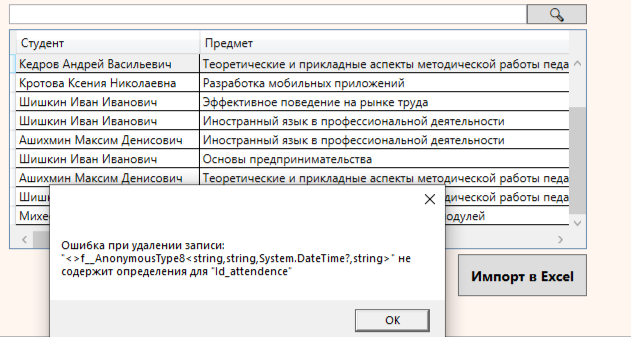


Рисунок 13 – Ошибка удаления записи

Ожидаемый результат: Ошибка о некорректных данных.

Полученный результат: Ошибка некорректных данных (рисунок 13).

Решение проблемы: На данный момент проблема с удалением не решена полностью.

* Попытка ввода неправильных данных при авторизации.

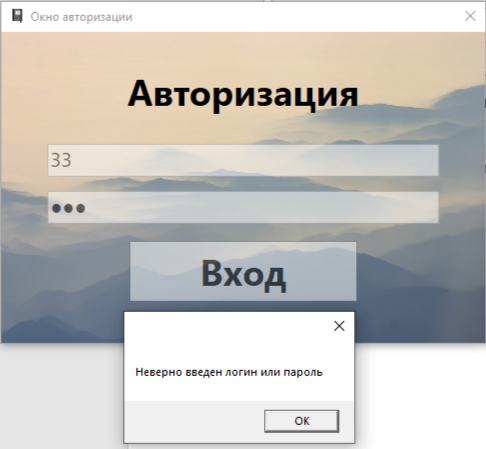


Рисунок 14 – Проверка правильности ввода

Ожидаемый результат: Ошибка о некорректных данных.

Полученный результат: Ошибка некорректных данных (рисунок 14).

Решение проблемы: На данный момент функция правильности ввода работает стабильно.

* Попытка поиска записей о посещаемости конкретного студента.

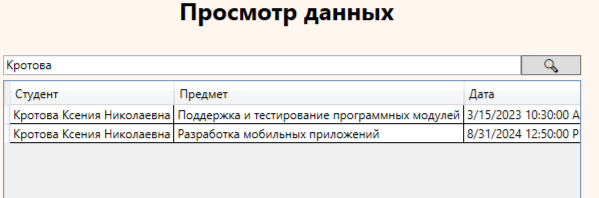


Рисунок 15 – Поиск записей

Ожидаемый результат: Получение правильных данных после нажатия кнопки «Поиск».

Полученный результат: Получение правильных данных (рисунок 15).

Решение проблемы: На данный момент функция поиска работает стабильно.

* Попытка сохранения записей о посещаемости в отчет.

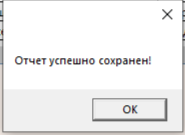


Рисунок 16 – Попытка сохранения записей

Ожидаемый результат: Сохранение данных после нажатия кнопки «Импорт в Excel».

Полученный результат: Сохранение данных (рисунок 16).

Решение проблемы: На данный момент функция сохранения в отчет работает стабильно.

* Попытка удаления пользователя.

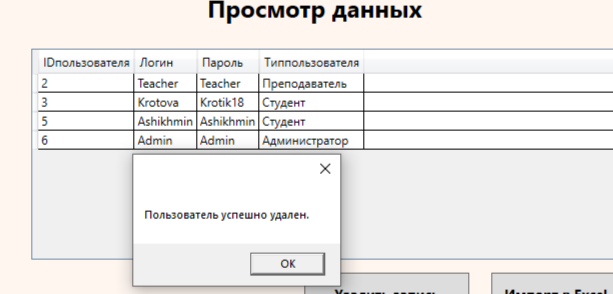


Рисунок 17 – Удаление пользователей

Ожидаемый результат: Удаление данных о пользователе после нажатия кнопки «Удалить запись».

Полученный результат: Успешное удаление данных (рисунок 16).

Решение проблемы: На данный момент функция удаления работает стабильно.

Тестирование программы осуществлялось на персональном компьютере со следующими техническими характеристиками:

* Процессор – Intel(R) Core(TM) i7-820Q CPU @ 1.73 GHz
* Оперативная память – DDR3 16 ГБ
* Видеокарта – AMD Radeon HD 6300 M series 512 МБ
* Операционная система – Windows 10

РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

Функциональным назначением программы является ведение и автоматизация учёта посещаемости учебных занятий студентами.

Программа должна обеспечивать возможность выполнения перечисленных ниже функций:

* Просмотр данных о группах, студентах, предметах, специальностях и учете посещаемости.
* Ведение учёта посредством добавления информации о пропусках занятий обучающимися.
* Возможность удаления, редактирования данных.
* Возможность сохранения данных в виде отчета.

Климатические условия эксплуатации, при которых должны обеспечиваться заданные характеристики, должны удовлетворять требованиям, предъявляемым к техническим средствам в части условий их эксплуатации.

В состав технических средств должен входить IBM-совместимый персональный компьютер (ПЭВМ), включающий себя:

* процессор с тактовой частотой, 1 ГГц, не менее;
* оперативную память объемом 512 Мб, не менее;
* жесткий диск со свободным местом 500 Мб, не менее;
* монитор, с разрешением экрана 1024\*768, не менее;
* оптический привод;
* компьютерная мышь;
* клавиатура;

Системные программные средства, используемые программой, должны быть представлены лицензионной локализованной версией операционной системы Windows 7/8/10/11.

Все пользователи должны обладать навыками работы с графическим пользовательским интерфейсом операционной системы.

Выполнение программы. Для запуска программного продукта необходимо запустить «DW» с расширением exe.

При запуске программы происходит отображение формы авторизации, на которой пользователю предлагается ввести логин и пароль от существующей учётной записи (рисунок 18).

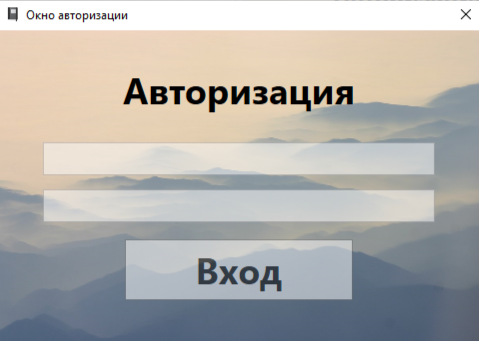


Рисунок 18 - Окно авторизации пользователя

В случае нажатия на кнопку «Вход» при корректном вводе данных, в зависимости от роли, назначенной пользователю в системе, откроется окно с соответствующими для пользователя возможностями.

Рассмотрим подробнее функционал преподавателя. Кнопка «Группы» запускает процесс показа списка групп, выгружая результаты на форму (рисунок 19).

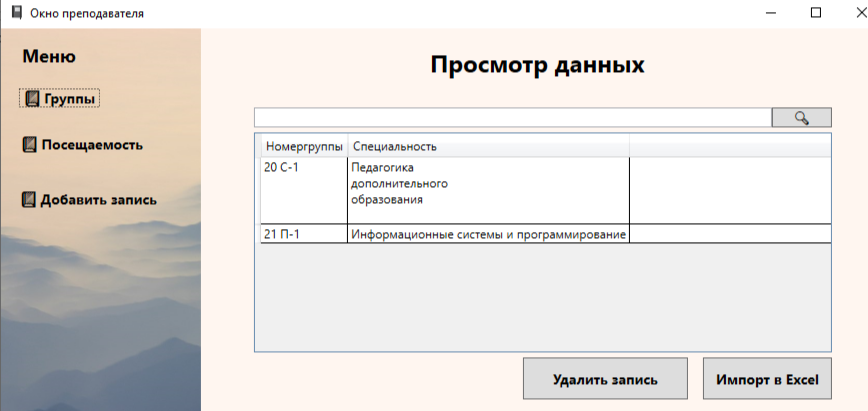


Рисунок 19 – Просмотр записей группах

Кнопка «Посещаемость» запускает процесс показа записей о пропусках, выгружая результаты на форму (рисунок 20).

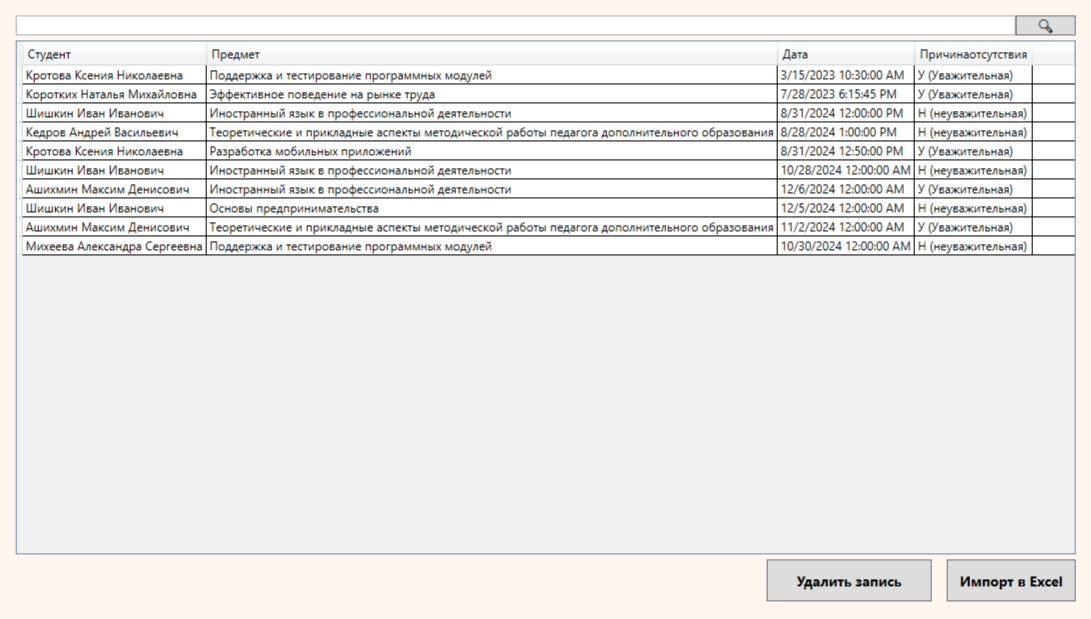


Рисунок 20 – Просмотр записей о посещаемости студентов

Кнопка «Импорт в Excel» запускает процесс сохранения данных о посещаемости студентов в файл xlsx (рисунок 21). Для работы с файлами Excel била использована библиотека EpPlus.

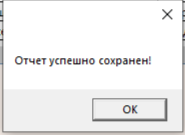


Рисунок 21 – Сохранение записей в отчет

Кнопка «Добавить запись» запускает новок окно для ввода данных о посещаемости студентов (рисунок 22). Все данные сохраняются корректно.

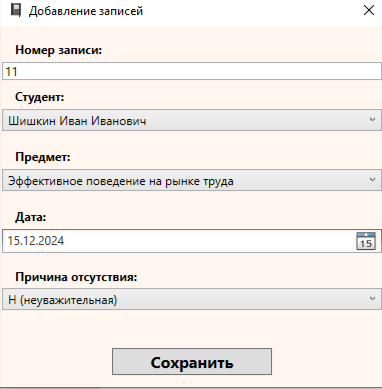


Рисунок 22 – Добавление записей о посещаемости

Кнопка «Поиск» дает возможность поиска записей о посещаемости конкретного студента (рисунок 23).

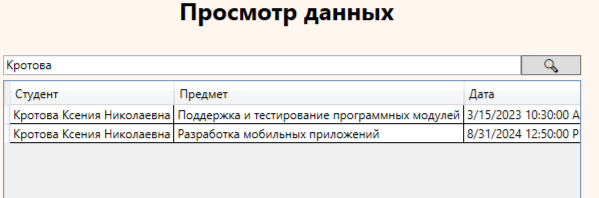


Рисунок 23 – Поиск записей

Были проверены все функциональные возможности программы при работе с реальными данными.

ЗАКЛЮЧЕНИЕ

Цель курсового проекта, которою являлось проектирование и создание программного модуля для учета пропусков учебных занятий выполнена полностью.

В ходе выполнения курсовой работы были реализованы следующие задачи:

• Проведён анализ предметной области;

• произведено создание технического задания на разработку программного модуля;

• описаны алгоритмы и функционирование программы.

В ходе проведения анализа предметной области были рассмотрены уже существующие программные решения под эту сферу деятельности, их достоинства и недостатки, на основании которых созданы критерии для проекта.

Создание технического задания было на основании рассмотренных ранее программ. Более конкретно определены критерии для программы, ее минимальные требования, доступность для различных операционных систем и конфигураций персональных компьютеров.

Функционал созданного программного решения расписан в описании алгоритмах и функционировании программы. Там подробно расписаны все функции программы, их мелочи, которые могут помочь при использовании данного приложения.

Код курсового проекта написан с использование стандартов кодинга. Элементы, переменные, модули и классы имеют название соответствующие значению этого элемента.

СПИСОК ЛИТЕРАТУРЫ

1. ГОСТ 19.101-77. Единая система программной документации. Виды программ и программных документов, введ. 01.01.1978. – г. Москва: Изд-во стандартов, 1980. – 4 с.
2. ГОСТ 19.201-78. Единая система программной документации. Техническое задание. Требования к содержанию и оформлению, введ. 01.01.1980. - г. Москва: Изд-во стандартов, 1988. - 3 с.
3. Агентство недвижимости. Введение, анализ предметной области [Электронный ресурс]. - <https://ru.wikipedia.org/wiki/Агентство_недвижимости>
4. Введение в .NET Framework. [Электронный ресурс]. - <https://ru.wikipedia.org/wiki/.NET_Framework>
5. Введение в WPF. [Электронный ресурс]. - <https://ru.wikipedia.org/wiki/Windows_Presentation_Foundation>
6. Документация по C#. Начало работы, руководства, справочная информация [Электронный ресурс]. – <https://learn.microsoft.com/ru-ru/dotnet/csharp/>
7. Документация по SQL. Начало работы, краткие руководства [Электронный ресурс]. – <https://learn.microsoft.com/RU-RU/sql/sql-server/educational-sql-resources?view=sql-server-ver15>
8. Сайт о программировании [Электронный ресурс]. – <https://metanit.com/>
9. Система вопросов и ответов о программировании [Электронный ресурс]. – <https://stackoverflow.com/>
10. Форум программистов [Электронный ресурс]. – <https://www.cyberforum.ru/>
11. Habr [Электронный ресурс]. – <https://habr.com/ru/>

ПРИЛОЖЕНИЯ

Приложение 1

**КОД ПРОГРАММЫ**

using System.Linq;

using System.Windows;

namespace Учет\_посещаемости

{

/// <summary>

/// Логика взаимодействия для MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

}

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

{

if (Log.Text == "" || Passw.Password == "")

{

MessageBox.Show("Заполните поля");

return;

}

using (var db = new Students\_absencesEntities())

{

var user = db.User.FirstOrDefault(u => u.Login == Log.Text && u.Password == Passw.Password);

if (user == null)

{

MessageBox.Show("Неверно введен логин или пароль");

}

else

{

MessageBox.Show("Успешная авторизация!");

var userType = db.User\_type.FirstOrDefault(t => t.Id\_types == user.Type\_User);

if (userType != null)

{

switch (userType.Type\_Name)

{

case "Admin":

GeneralWindow adminWindow = new GeneralWindow();

adminWindow.Show();

Close();

break;

case "Teacher":

TeacherWindow teacherWindow = new TeacherWindow();

teacherWindow.Show();

Close();

break;

case "Student":

StudentWindow studentWindow = new StudentWindow(user.Id\_student);

studentWindow.Show();

Close();

break;

default:

MessageBox.Show("Неизвестный тип пользователя");

break;

}

}

else

{

MessageBox.Show("Ошибка получения типа пользователя");

}

}

}

}

}

}

using System.Linq;

using System.Windows;

namespace Учет\_посещаемости

{

/// <summary>

/// Логика взаимодействия для StudentWindow.xaml

/// </summary>

public partial class StudentWindow : Window

{

private int \_currentUserId;

public StudentWindow(int? currentUserId)

{

InitializeComponent();

\_currentUserId = (int)currentUserId;

if (currentUserId.HasValue)

{

using (var db = new Students\_absencesEntities())

{

var user = db.User.FirstOrDefault(u => u.Id\_student == currentUserId.Value);

if (user != null)

{

var student = db.Student.FirstOrDefault(s => s.Id\_student == user.Id\_student);

if (student != null)

{

Info.Content = $"{student.Name} {student.Surname}";

}

}

}

}

}

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

{

using (var bd = new Students\_absencesEntities())

{

var Grup = from z in bd.Group

join sz in bd.Specialization on z.Specialization equals sz.Id\_specialization

select new

{

Номер\_группы = z.Id\_group + " " + z.Group\_Name,

Специальность = sz.Specialization\_Name

};

grid.ItemsSource = Grup.ToList();

}

}

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

{

using (var bd = new Students\_absencesEntities())

{

var Predm = from z in bd.Subject

select new

{

Наименование\_предмета = z.Name

};

grid.ItemsSource = Predm.ToList();

}

}

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

{

using (var bd = new Students\_absencesEntities())

{

var Cpez = from z in bd.Specialization

select new

{

Номер\_специальности = z.Number\_Specialization,

Наименование\_специальности = z.Specialization\_Name

};

grid.ItemsSource = Cpez.ToList();

}

}

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

{

using (var bd = new Students\_absencesEntities())

{

var Attend = from z in bd.Attendance

join sz in bd.Student on z.Id\_student equals sz.Id\_student

join pz in bd.Subject on z.Subject equals pz.Id\_subject

select new

{

Студент = sz.Surname + " " + sz.Name + " " + sz.Middle\_Name,

Предмет = pz.Name,

Дата = z.Date,

Причина\_отсутствия = z.Reason

};

grid.ItemsSource = Attend.ToList();

}

}

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

{

MainWindow mw = new MainWindow();

mw.Show();

this.Close();

}

}

}

using System.Linq;

using System.Windows;

using OfficeOpenXml;

using System.IO;

using System;

using Microsoft.Win32;

using System.Windows.Controls;

namespace Учет\_посещаемости

{

/// <summary>

/// Логика взаимодействия для TeacherWindow.xaml

/// </summary>

public partial class TeacherWindow : Window

{

public TeacherWindow()

{

InitializeComponent();

}

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

{

MainWindow mw = new MainWindow();

mw.Show();

this.Close();

}

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

{

using (var bd = new Students\_absencesEntities())

{

var Grup = from z in bd.Group

join sz in bd.Specialization on z.Specialization equals sz.Id\_specialization

select new

{

Номер\_группы = z.Id\_group + " " + z.Group\_Name,

Специальность = sz.Specialization\_Name

};

grid.ItemsSource = Grup.ToList();

}

}

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

{

using (var bd = new Students\_absencesEntities())

{

var Attend = from z in bd.Attendance

join sz in bd.Student on z.Id\_student equals sz.Id\_student

join pz in bd.Subject on z.Subject equals pz.Id\_subject

select new

{

Студент = sz.Surname + " " + sz.Name + " " + sz.Middle\_Name,

Предмет = pz.Name,

Дата = z.Date,

Причина\_отсутствия = z.Reason

};

grid.ItemsSource = Attend.ToList();

}

}

private void New\_Pos\_Click(object sender, RoutedEventArgs e)

{

AddAttendance mw = new AddAttendance();

mw.Show();

this.Hide();

}

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

{

string searchQuery = SearchBox.Text;

using (var bd = new Students\_absencesEntities())

{

var studentSearch = from z in bd.Student

join attend in bd.Attendance on z.Id\_student equals attend.Id\_student

join subject in bd.Subject on attend.Subject equals subject.Id\_subject

where (z.Surname + " " + z.Name + " " + z.Middle\_Name).Contains(searchQuery)

select new

{

Студент = z.Surname + " " + z.Name + " " + z.Middle\_Name,

Предмет = subject.Name,

Дата = attend.Date,

Причина\_отсутствия = attend.Reason

};

var results = studentSearch.ToList();

grid.ItemsSource = results;

}

}

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

{

using (var bd = new Students\_absencesEntities())

{

var Attend = from z in bd.Attendance

join sz in bd.Student on z.Id\_student equals sz.Id\_student

join pz in bd.Subject on z.Subject equals pz.Id\_subject

select new

{

Студент = sz.Surname + " " + sz.Name + " " + sz.Middle\_Name,

Предмет = pz.Name,

Дата = z.Date,

Причина\_отсутствия = z.Reason

};

var dataToExport = Attend.ToList();

SaveFileDialog saveFileDialog = new SaveFileDialog();

saveFileDialog.Filter = "Excel Files|\*.xlsx";

saveFileDialog.Title = "Сохранить отчет";

saveFileDialog.FileName = "Отчет";

if (saveFileDialog.ShowDialog() == true)

{

string filePath = saveFileDialog.FileName;

using (ExcelPackage excelPackage = new ExcelPackage())

{

var worksheet = excelPackage.Workbook.Worksheets.Add("Отчет");

worksheet.Cells[1, 1].Value = "Студент";

worksheet.Cells[1, 2].Value = "Предмет";

worksheet.Cells[1, 3].Value = "Дата";

worksheet.Cells[1, 4].Value = "Причина отсутствия";

for (int i = 0; i < dataToExport.Count; i++)

{

worksheet.Cells[i + 2, 1].Value = dataToExport[i].Студент;

worksheet.Cells[i + 2, 2].Value = dataToExport[i].Предмет;

worksheet.Cells[i + 2, 3].Value = dataToExport[i].Дата;

worksheet.Cells[i + 2, 4].Value = dataToExport[i].Причина\_отсутствия;

}

worksheet.Column(3).Style.Numberformat.Format = "dd.MM.yyyy";

FileInfo excelFile = new FileInfo(filePath);

excelPackage.SaveAs(excelFile);

}

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

}

grid.ItemsSource = dataToExport;

}

}

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

{

try

{

if (grid.SelectedItem == null)

{

MessageBox.Show("Пожалуйста, выберите запись для удаления.");

return;

}

var selectedAttendance = grid.SelectedItem as dynamic;

if (selectedAttendance != null)

{

int attendanceId = selectedAttendance.Id\_attendence;

using (var bd = new Students\_absencesEntities())

{

var attendanceRecord = bd.Attendance.FirstOrDefault(a => a.Id\_attendence == attendanceId);

if (attendanceRecord != null)

{

bd.Attendance.Remove(attendanceRecord);

bd.SaveChanges();

MessageBox.Show("Запись успешно удалена!");

}

else

{

MessageBox.Show("Запись не найдена.");

}

}

}

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка при удалении записи: {ex.Message}");

}

}

}

}

using System;

using System.Linq;

using System.Windows;

namespace Учет\_посещаемости

{

/// <summary>

/// Логика взаимодействия для AddAttendance.xaml

/// </summary>

public partial class AddAttendance : Window

{

public AddAttendance()

{

InitializeComponent();

LoadStudents();

LoadSubjects();

LoadReason();

}

private void LoadStudents()

{

using (var bd = new Students\_absencesEntities())

{

var students = bd.Student.Select(s => new { s.Id\_student, FullName = s.Surname + " " + s.Name + " " + s.Middle\_Name }).ToList();

Fio.ItemsSource = students;

Fio.DisplayMemberPath = "FullName";

Fio.SelectedValuePath = "Id\_student";

}

}

private void LoadSubjects()

{

using (var bd = new Students\_absencesEntities())

{

var subjects = bd.Subject.Select(s => new { s.Id\_subject, s.Name }).ToList();

Predmet.ItemsSource = subjects;

Predmet.DisplayMemberPath = "Name";

Predmet.SelectedValuePath = "Id\_subject";

}

}

private void LoadReason()

{

using (var bd = new Students\_absencesEntities())

{

var reasons = (from a in bd.Attendance

select new

{

Display = a.Reason,

Value = a.Id\_attendence

}).Distinct().ToList();

Prich.ItemsSource = reasons;

Prich.DisplayMemberPath = "Display";

Prich.SelectedValuePath = "Value";

}

}

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

{

if (Fio.SelectedValue == null || Predmet.SelectedValue == null || Dat.SelectedDate == null || Prich.SelectedValue == null)

{

MessageBox.Show("Пожалуйста, заполните все поля.");

return;

}

using (var bd = new Students\_absencesEntities())

{

var newAttendance = new Attendance

{

Id\_attendence = Convert.ToInt32(Nom.Text),

Id\_student = (int)Fio.SelectedValue,

Subject = (int)Predmet.SelectedValue,

Date = Dat.SelectedDate.Value,

Reason = Prich.Text

};

bd.Attendance.Add(newAttendance);

bd.SaveChanges();

MessageBox.Show("Изменения успешно сохранены!");

TeacherWindow mw = new TeacherWindow();

mw.Show();

this.Close();

}

}

}

}

Приложение 2

USB-накопитель с материалами проекта

На USB-накопителе располагается:

* Установщик программы
* Проект программы
* Файл курсового проекта в формате MS Word
* Файл с презентацией курсового проекта