Министерство образования и науки Нижегородской области

Государственное бюджетное профессиональное   
образовательное учреждение

«Заволжский автомоторный техникум»

(ГБПОУ ЗАМТ)

Специальность: 09.02.03 «Программирование в компьютерных системах»

**Выпускная квалификационная работа**

Студента IV курса группы ПС-20Б

**Костина Никиты Константиновича**

**Тема:** Разработка автоматизированной информационной системы учета посещаемости занятий на базе платформы .NET

Число листов ВКР (прописью) – \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Число страниц пояснительной записки –\_\_\_\_\_\_ страниц

|  |  |
| --- | --- |
| **Допустить к защите:**  Зам. директора по УР  Мокрышева Л.Г.  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) | **Руководитель**  выпускной квалификационной работы:  Хмелева Е.О.  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись)  **Рецензент:**  Лебедев В.Л.  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) |

Работа защищена \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата)

г. Заволжье

2024 г.

Министерство образования и науки Нижегородской области

Государственное бюджетное профессиональное   
образовательное учреждение

«Заволжский автомоторный техникум»

(ГБПОУ ЗАМТ)

Специальность: 09.02.03 «Программирование в компьютерных системах»

Утверждаю

Зам. директора по УР

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Мокрышева Л.Г

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 год

**задание**

**на выпускную квалификационную работу**

Студента IV курса группы ПС-20Б

**Костина Никиты Константиновича**

**Тема выпускной квалификационной работы:**

Разработка автоматизированной информационной системы учета посещаемости занятий на базе платформы. NET

Выпускная квалификационная работа на указанную тему выполняется в следующем объеме:

### Пояснительная записка

Введение

1 Постановка задачи

2 Проектирование

2.1 Обоснование выбора среды разработки

2.2 Выбор способа организации входных и выходных данных

2.3 Проектирование моделей данных

2.4 Алгоритм представления задачи

2.4.1 Детальное проектирование программного обеспечения (прототип)

2.4.2 Описание основных программных модулей

2.5 Применяемые меры защиты информации

3 Тестирование

4 Экономическое обоснование

4.1 Расчет затрат на разработку программы

4.2 Стратегия продвижения программного продукта

4.3 Расчет цены разработанной программы

4.4 Технико-экономические показатели разработки программы

Заключение

Литература

Приложения

#### Дата выдачи задания « 13» \_\_04\_\_\_\_ 2024 года

#### Дата окончания проекта « 14 » \_\_\_06\_\_ 2024 года

Руководитель ВКР \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ Хмелева Е.О./

Председатель ПЦК \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Карпушева И.А./

Содержание

Введение 4

1 Постановка задачи 5

2 Проектирование 7

2.1 Обоснование выбора среды разработки 7

2.2 Выбор способа организации входных и выходных данных 9

2.3 Проектирование моделей данных 14

2.4 Алгоритм представления задачи 17

2.4.1 Детальное проектирование программного обеспечения (прототип) 17

2.4.2 Описание основных программных модулей 25

2.5 Применяемые меры защиты информации 27

3 Тестирование 31

4 Экономическое обоснование 37

4.1 Расчет затрат на разработку программы 39

4.2 Стратегия продвижения программного продукта 41

4.3 Расчет цены разработанной программы 42

4.4 Технико-экономические показатели разработки программы 43

Заключение 44

Литература 45

Приложения

Введение

В век информационных технологий важнейшей характеристикой для каждого человека становится скорость доступа к нужной информации. Одной из таких является информация о посещаемости занятий внутри техникума. Для получения этой информации и начинают вести учет.

Проблема такого учета состоит в том, что как правило он ведется в бумажном виде, что может вызывать сложности с доступом к данным, риск потери той или иной информации, а также необходимость ручного обновления и проверки документов.

Целью выпускной квалификационной работы является - разработка автоматизированной информационной системы учета посещаемости занятий на базе платформы .NET.

В процессе проектирования необходимо решить следующие задачи:

- сформулировать требования к программе;

- построить структуру базы данных;

- выбрать среду разработки;

- реализовать программный продукт;

- протестировать программный продукт;

- рассчитать технико-экономические показатели;

- разработать техническую и эксплуатационную документацию.

Автоматизированная информационная система позволит автоматизировать учет посещаемости. Староста сможет вести учет. Куратор назначать старосту, просматривать посещаемость и составлять по ней отчеты. Работник учебной части сможет назначать кураторов из преподавателей. А администратор сможет быстро подготовить систему к работе с помощью функции импорта, получит возможность управлять жизненным циклом учетных записей (создавать, редактировать и удалять их), а также управлять резервным копированием и восстановлением данных системы.

1. **Постановка задачи**

Разработать и реализовать автоматизированную информационную систему учета посещаемости занятий. В программе должны соблюдаться принципы информационной закрытости, модульности и иерархической организации.

**Предметная область**: учет посещаемости занятий содержит необходимую информацию о студентах и их пропусках, а также о группах их кураторах и старост.

Автоматизированная информационная система учета посещаемости занятий предназначена для ведения базы данных пропусков студентов, а также базы данных пользователей системы, сотрудников учебной части, преподавателей, групп и студентов.

Автоматизированная информационная система учета посещаемости занятий должна иметь форму авторизации (администратор, сотрудник учебной части, куратор группы, староста группы), в режиме администратора должна быть предусмотрена возможность добавления нового пользователя.

Автоматизированная информационная учета посещаемости занятий должна иметь возможность добавления сотрудников, преподавателей, групп и самих учащихся в систему учета, назначение кураторов и старост группы, а также ведения учета посещаемости занятий, то есть занесении информации о пропуске занятий.

Автоматизированная информационная система учета посещаемости занятий должна иметь возможность ведения и редактирования базы данных (добавление, удаление, редактирование, сохранение).

Пользователь должен иметь возможность отсортировать информацию (по возрастанию и убыванию) по разным параметрам. Возможность отфильтровать данные. Реализовать живой поиск информации.

Предусмотреть возможность составления отчетов.

Автоматизированная информационная система учета посещаемости занятий должна иметь возможность импорта и экспорта данных базы в CSV.

Язык программирования С#. База данных составляется в MySQL.

*Администратор АИС имеет возможность:*

1. Выполнить первоначальную настройку АИС (выполнить подключение к БД, добавить первого пользователя – сотрудника учебной части);
2. заполнять справочники с помощью импорта данных;
3. восстанавливать базу данных.

*Сотрудник учебной части имеет возможность:*

1. Создавать нового сотрудника учебной части, как пользователя системы;

*Куратор группы имеет возможность:*

1. Создавать нового старосту для своей группы, как пользователя системы;
2. просматривать посещаемость занятий своей группы;
3. создавать и экспортировать отчеты посещаемости;

*Староста группы имеет возможность:*

1. Вести учет посещаемости занятий своей группы;

**2 Проектирование**

2.1 Обоснование выбора среды разработки

Для создания данного продукта можно использовать следующие среды разработки: Microsoft Visual Studio версия Community, Rider и [Eclipse](https://www.eclipse.org/).

*Microsoft Visual Studio Community* – бесплатная интегрированная среда разработки (IDE) от Microsoft, предназначенная для создания различных приложений, включая веб-сайты, мобильные приложения, игры и многое другое. Программа поддерживает различные языки программирования, включая C#, Visual Basic, F#, JavaScript, Python и другие. Visual Studio Community обладает широким набором инструментов для удобной разработки, отладки и тестирования кода, а также расширяем через множество плагинов и расширений.

[*Rider*](https://code.visualstudio.com/)– это интегрированная среда разработки (IDE) от компании JetBrains, предназначенная для создания приложений на платформах .NET и .NET Core. Rider поддерживает различные языки программирования, включая C#, F#, Visual Basic, JavaScript, TypeScript, HTML и CSS. Программа обладает рядом полезных функций, таких как интеллектуальное автодополнение кода, интеграция с системами управления версиями, встроенные отладчики и возможность проведения юнит-тестирования, распространяется по лицензии.

Eclipse – это бесплатная open-sourse интегрированная среда разработки (IDE), предназначенная для создания приложений на различных языках программирования, включая Java, C/C++, Python, PHP и другие. Имеет большой функционал благодаря множеству плагинов, но при этом имеет достаточно низкую производительность по сравнению с другими аналогичными программами, также, из-за того, что программа является open-sourse, не редко возникают ошибки в работе программы.

Исходя из вышеперечисленных инструментальных средств, была выбрана среда разработки Microsoft Visual Studio Community, так как данная среда разработки имеет достаточный функционал для написания данного проекта и является бесплатной при разработке некоммерческих программных продуктов. Также имеется опыт работы при создании подобных проектов, что значительно сократит время разработки.

Данная среда разработки позволяет программировать на следующих языках программирования: С#, C++, Visual Basic и другие. Если необходимого языка нет в стандартной библиотеке, его можно внедрить без особых трудностей.

*C# -* современный объектно-ориентированный язык программирования, разработанный компанией Microsoft, используется для создания разнообразных приложений, включая приложения для Windows, веб-приложения, сервисы и игры. Интеграция с платформой .NET обеспечивает высокую совместимость с другими технологиями и инструментами от Microsoft.

*Visual Basic* – высокоуровневый язык программирования, разработанный компанией Microsoft. Он позволяет разрабатывать графические пользовательские интерфейсы и приложения для операционных систем Windows. Visual Basic обладает простым и понятным синтаксисом, что делает его доступным для начинающих программистов, но имеет ограниченные возможности для разработки кросс-платформенных приложений, а так же достаточно низкую производительность, по сравнению с другими аналогичными языками программирования.

*C++* - высокоуровневый язык программирования, который широко используется для разработки системного и прикладного программного обеспечения, имеет статическую типизацию, обширные возможности работы с памятью и поддерживает объектно-ориентированное программирование, обладает высокой производительностью, но из-за необходимости внимательного контроля и управления памятью, возможны утечки памяти и ошибки в программе.

Исходя из выше перечисленных языков программирования для разработки был выбран язык С#, так как это объектно-ориентированный высокоуровневый язык программирования, то есть он подходит для написания графических приложений. Работа с внутренностями .Net не требуется, сроки решения задачи не допускают этого, гораздо больше внимания следует уделить непосредственно задаче, а не техническим деталям. Исходя из этого, предпочтение было отдано в сторону С#, так как исползование C++ для выполнения выпускной квалификационной работы нерационально, а *Visual Basic* не соответствует постановке задачи.

2.2 Выбор способа организации входных и выходных данных

Для организации входных и выходных данных необходима база данных. Visual Studio позволяет использовать базы данных: MS Access, SQLite, MуSQL и другие.

PostgreSQL – это свободная реляционная система управления базами данных, разработанная для обеспечения надежности, масштабируемости и производительности, хорошо работает с большими базами данных, но потребляет много системных ресурсов, так же имеет довольно слабый базовый функционал в сравнении с аналогами.

*MySQL* - это свободная реляционная база данных, которая используется для хранения структурированных данных, имеет открытый исходный код, высокую производительность, надежность, масштабируемость и поддержку стандартного языка запросов SQL. Кроме того, MySQL имеет обширное сообщество пользователей и разработчиков, что облегчает получение необходимой поддержки и дополнительных ресурсов.

*SQLite* – это легковесная база данных, которая обеспечивает хранение данных в локальном файле без необходимости установки сервера. Она поддерживает стандартный SQL, обеспечивает высокую производительность и надежность работы, но не поддерживает многопользовательский доступ и плохо подходит для работы с большими базами данных.

Исходя из выше перечисленных вариантов, принято решение использовать реляционную систему управления базами данных *MySQL*, так как ее функционала хватает для выполнения поставленной задачи, имеется практический навык, позволит в песпективе перейти к клиент-серверной версии.

База данных состоит из 10 таблиц, которые имеют связь в соответствии с рисунком 2.1

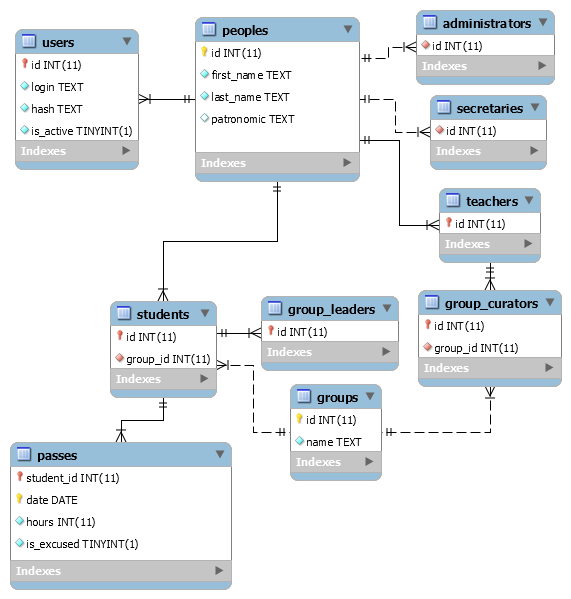


Рисунок 2.1 – Структура базы данных

Структура таблиц базы данных представлена таблицами 2.1-2.10

Таблица 2.1 – Структура таблицы «peoples» (Люди)

|  |  |  |  |
| --- | --- | --- | --- |
| Название столбца | Название столбца  (в БД) | Тип данных | Ограничения |
| Идентификатор человека | id | Счетчик | длинное целое число, заполняется автоматически |
| Имя | first\_name | Текстовый |  |
| Фамилия | last\_name | Текстовый |  |
| Отчество | patronomic | Текстовый |  |

Таблица 2.2 – Структура таблицы «users» (Пользователи)

|  |  |  |  |
| --- | --- | --- | --- |
| Название столбца | Название столбца (в БД) | Тип данных | Ограничения |
| Идентификатор человека | id | Счетчик | длинное целое число, ссылается на человека |
| Логин | login | Текстовый |  |
| Хэш пароля | hash | Текстовый |  |
| Активен | is\_active | Логическое значение |  |

Таблица 2.3 – Структура таблицы «administrators» (Администраторы)

|  |  |  |  |
| --- | --- | --- | --- |
| Название столбца | Название столбца (в БД) | Тип данных | Ограничения |
| Идентификатор человека | id | Счетчик | длинное целое число, ссылается на человека |

Таблица 2.4 – Структура таблицы «secretaries» (Работники учебной части)

|  |  |  |  |
| --- | --- | --- | --- |
| Название столбца | Название столбца (в БД) | Тип данных | Ограничения |
| Идентификатор человека | id | Счетчик | длинное целое число, ссылается на человека |

Таблица 2.5 – Структура таблицы «teachers» (Преподаватели)

|  |  |  |  |
| --- | --- | --- | --- |
| Название столбца | Название столбца (в БД) | Тип данных | Ограничения |
| Идентификатор человека | id | Счетчик | длинное целое число, ссылается на человека |

Таблица 2.6 – Структура таблицы «groups» (Группы)

|  |  |  |  |
| --- | --- | --- | --- |
| Название столбца | Название столбца (в БД) | Тип данных | Ограничения |
| Идентификатор группы | id | Счетчик | длинное целое число, заполняется автоматически |
| Наименование группы | name | Текстовый |  |

Таблица 2.7 – Структура таблицы «group\_curators» (Кураторы групп)

|  |  |  |  |
| --- | --- | --- | --- |
| Название столбца | Название столбца (в БД) | Тип данных | Ограничения |
| Идентификатор преподавателя | id | Счетчик | длинное целое число, ссылается на преподавателя |
| Идентификатор группы | group\_id | Счетчик | длинное целое число, ссылается на группу |

Таблица 2.8 – Структура таблицы «students» (Студенты)

|  |  |  |  |
| --- | --- | --- | --- |
| Название столбца | Название столбца (в БД) | Тип данных | Ограничения |
| Идентификатор человека | id | Счетчик | длинное целое число, ссылается на человека |
| Идентификатор группы | group\_id | Счетчик | длинное целое число, ссылается на группу |

Таблица 2.9 – Структура таблицы «group\_leaders» (Старосты групп)

|  |  |  |  |
| --- | --- | --- | --- |
| Название столбца | Название столбца (в БД) | Тип данных | Ограничения |
| Идентификатор студента | id | Счетчик | длинное целое число, ссылается на студента |

Таблица 2.10 – Структура таблицы «passes» (Пропуски)

|  |  |  |  |
| --- | --- | --- | --- |
| Название столбца | Название столбца (в БД) | Тип данных | Ограничения |
| Идентификатор студента | student\_id | Счетчик | длинное целое число, ссылается на студента |
| Дата пропуска | date | Дата |  |
| Количество пропущенных часов | hours | Число |  |
| Уважителен | is\_excused | Логическое значение |  |

2.3 Проектирование моделей данных

При проектировании моделей данных были разработаны диаграммы: вариантов использования и деятельности.

Диаграммы вариантов использования применяются при бизнес – анализе для моделирования видов работ, выполняемых организацией, и для моделирования функциональных требований к ПС при ее проектировании и разработке.

Диаграмма вариантов использования описывает функциональное назначение системы, т.е. что система будет делать в процессе своего функционирования. Является исходной концептуальной моделью системы в процессе ее проектирования и разработки. Диаграмма вариантов использования cсистемы автоматизированной информационной системы по учету заказов мебельного магазина представленная на рисунке 2.2



Рисунок 2.2 - Диаграмма вариантов использования

Сценарий использования – вариант использования, прецедент или же пользовательский сценарий в разратке программного обеспечения и системном проектировании. Это описание поведения системы, которым она отвечает на внешние запросы. Другими словами, сценарий использования описывает, «кто» и «что» может сделать с рассматриваемой системой. Методика сценариев использования применяется для выявления требований к поведению системы, известных также как функциональные требования. Сценарий диаграммы представлен в таблице 2.8

Таблица 2.8 Сценарий вариантов использования

|  |  |
| --- | --- |
| Действие актеров | Отклик |
| 1 | 2 |
| Варианты использования | Формирование отчета по посещаемости за семестр |
| Актеры | Куратор |
| Краткое описание | Вход в АИС в режиме «Куратор». Открыть пункт «Посещаемость за период» в главном меню, выбрать период, нажать кнопку «Экспортировать отчет». |
| Цель | Занесение информации в отчет для подготовки его к печати. |
| Тип | Базовый |
| Ссылки на другие варианты использования | - экспорт отчетов; |

Актер это кто-то или что-то вне системы и влияющий на систему или находящийся под ее влиянием. Актер может быть человеком, устройством, другой системой или подсистемой, или временем. Человек в реальном мире может быть представлен несколькими актерами, если у них есть несколько различных ролей и целей по отношению к системе. Они взаимодействуют с системой и производят над ней некоторые действия.

Краткое описание. Варианты применения сценариев в процессе разработки зависят от используемой методологии разработки. В одних методологиях разработки все, что требуется - это краткий обзор сценария.

Цель – решение поставленных задач. Нет никакой необходимости в сценарии использования, когда нет никакой потребности ни в каком актере, чтобы достигнуть цели. Цель кратко описывает то, чего пользователь намеревается достигнуть с этим сценарием, в соответствии с таблицей 2.9

Таблица 2.9 Последовательность действий актеров

|  |  |
| --- | --- |
| Действия актеров | Отклик |
| 1 | 2 |
| 1. Куратор запускает программу | 1. Открытие формы авторизации |
| 1. Куратор вводит логин и пароль | 1. Открытие главного меню |
| 1. Куратор переходит на форму просмотра посещаемости за период | 1. Оторбражаются данные о посещаемости по его группе |
| 1. Куратор выбирает период | 1. Отображаются данные за период |
| 1. Нажатие на кнопку «Экспортировать отчет» | 1. Экспот отчета по посещаемости в MS Excel 2007 |

Диаграмма деятельности – UML-диаграмма, на которой показано разложение некоторой деятельности на её составные части. Под деятельностью понимается спецификация исполняемого поведения в виде координированного последовательного и параллельного выполнения подчиненных элеемнтов – вложенных видов деятельности и отдельных действий, соединенных между собой потоками, которые идут от выходов одного узла к входам другого.

Диаграмма деятельности используется при моделированиии бизнес-процессов, технологических процессов, последовательных и параллельных вычислений. Диаграмма деятельности программного продукта продемонстрирована на рисунке 2.3

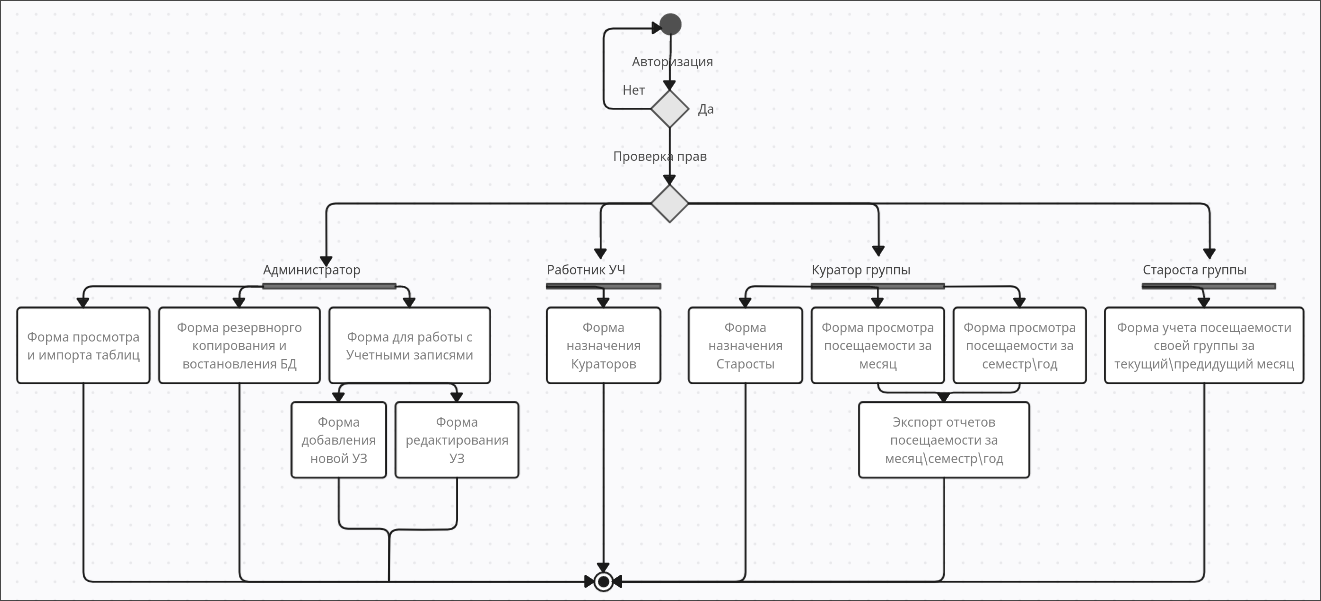


Рисунок 2.3 - Диаграмма деятельности

2.4 Алгоритм представления задачи

2.4.1 Детальное проектирование пользовательского интерфейса

Детальное проектирование ПО – это подробное описание компонентов программного обеспечения и интерфейсов между ними.

Прототипирование программного обеспечения – этап разработки программного обеспечения, процесс создания прототипа программы – макета (черновой, пробной версии) программы, обычно – с целью проверки пригодности предлагаемых для применения концепций, архитектурных или технологических решений, а также для представления программы заказчику на ранних стадиях процесса разработки.

На рисунке 2.4 показаны формы аторизации

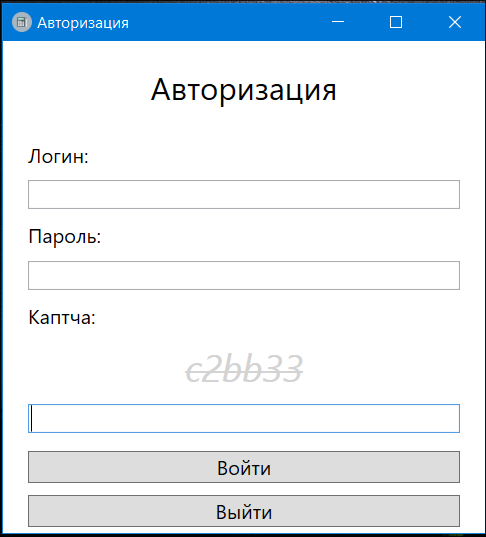
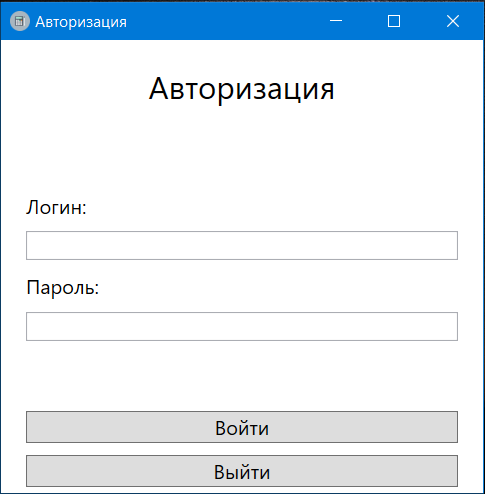


Рисунок 2.4 – Формы авторизации

На рисунках 2.5 – 2.8 показано главное меню

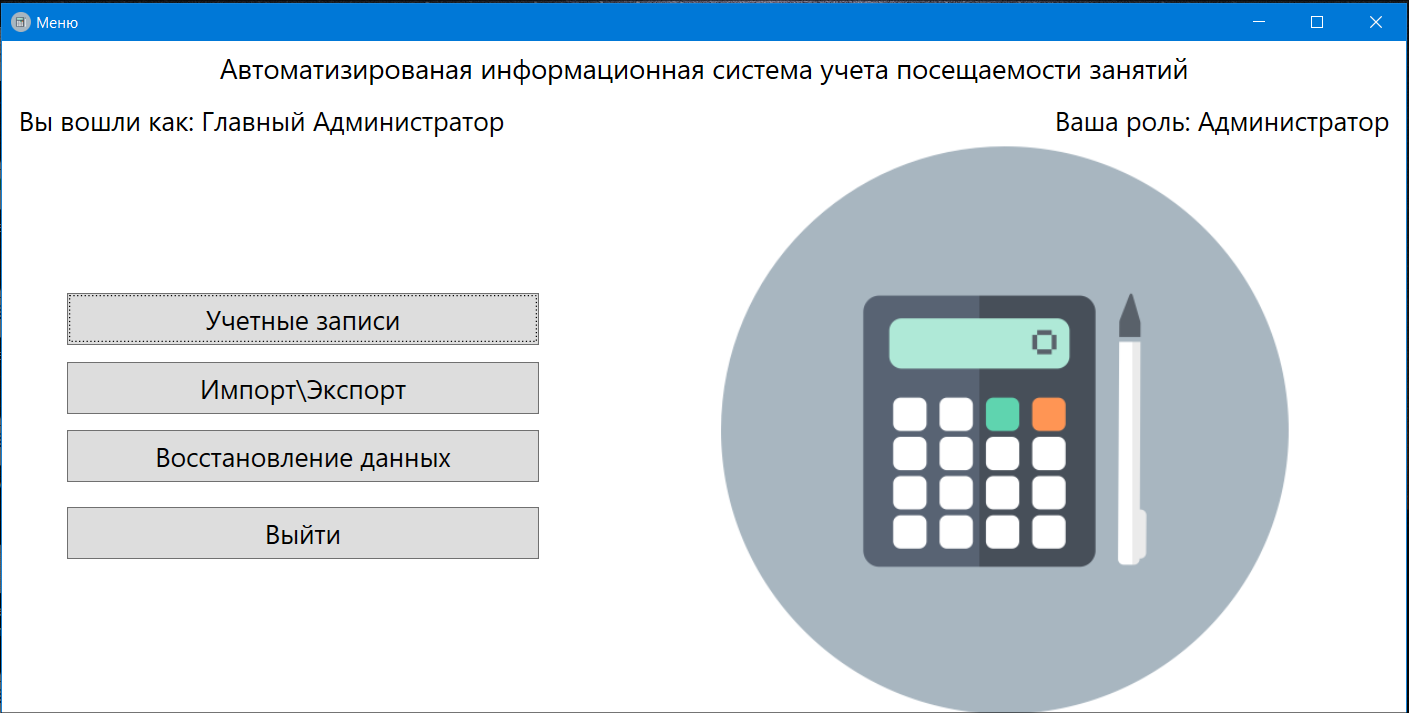


Рисунок 2.5 – Форма меню (администратора)

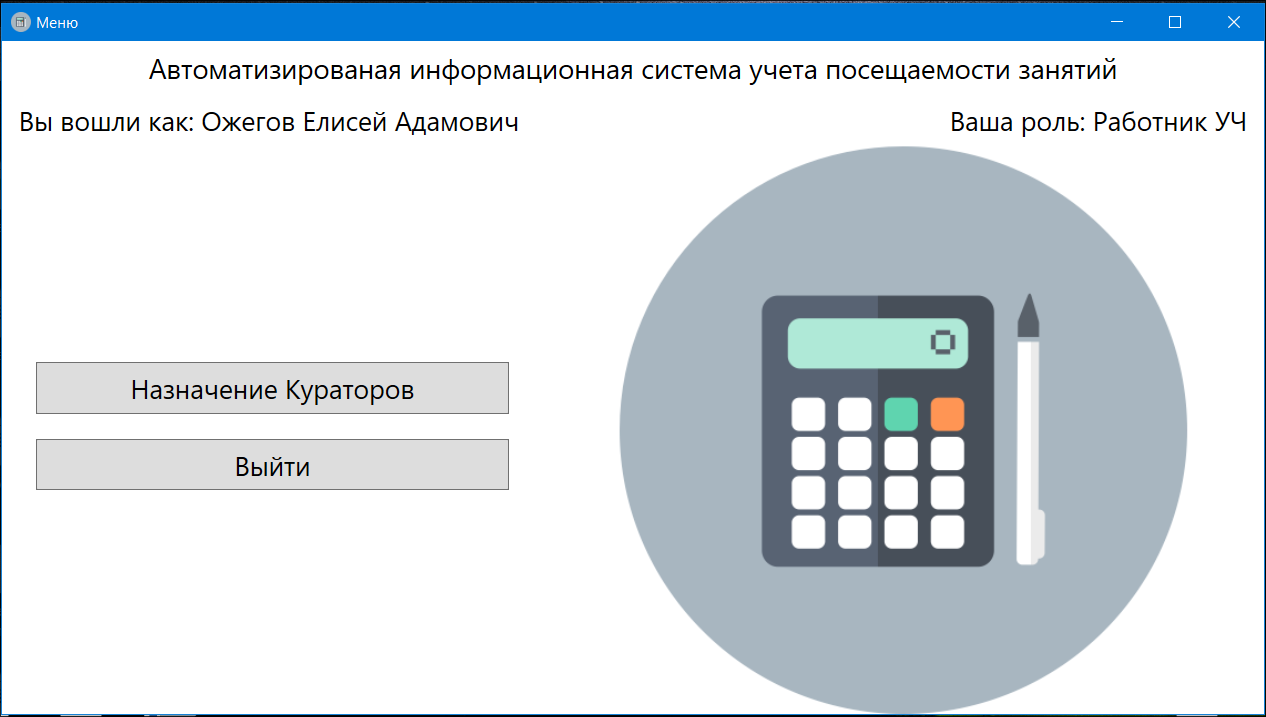


Рисунок 2.6 – Форма меню (работника учебной части)

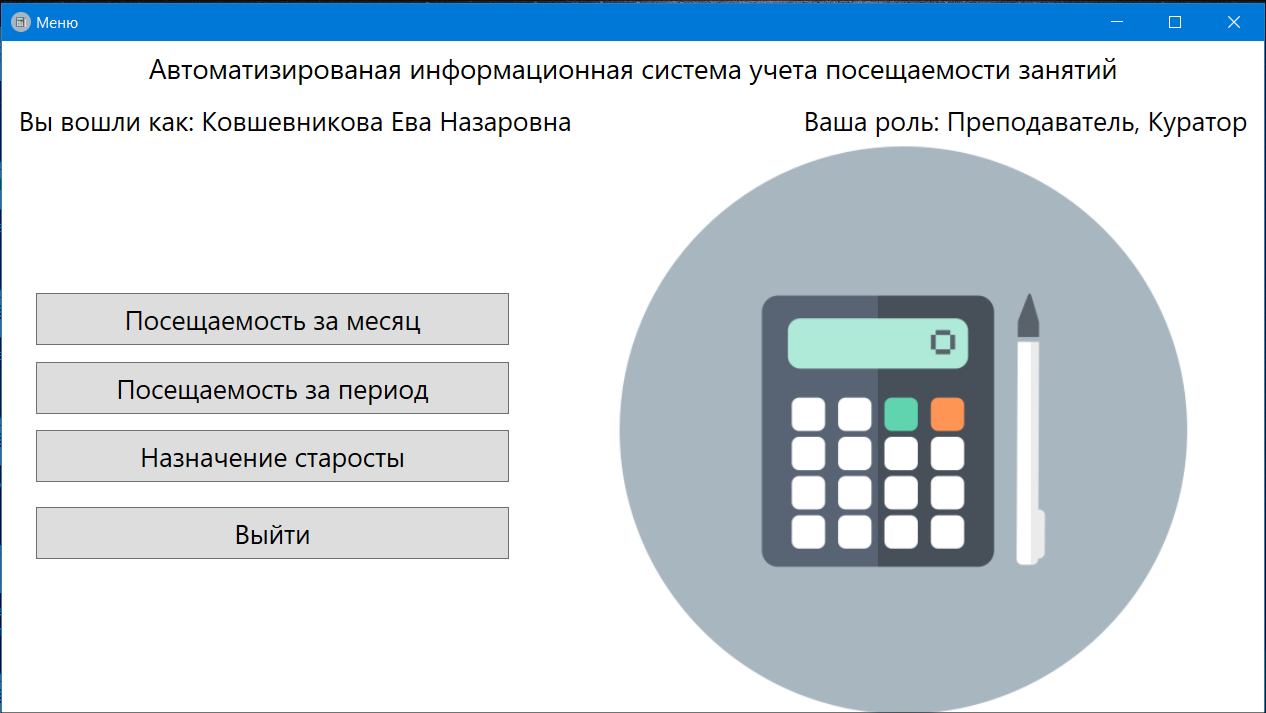


Рисунок 2.7 – Форма меню (куратора)

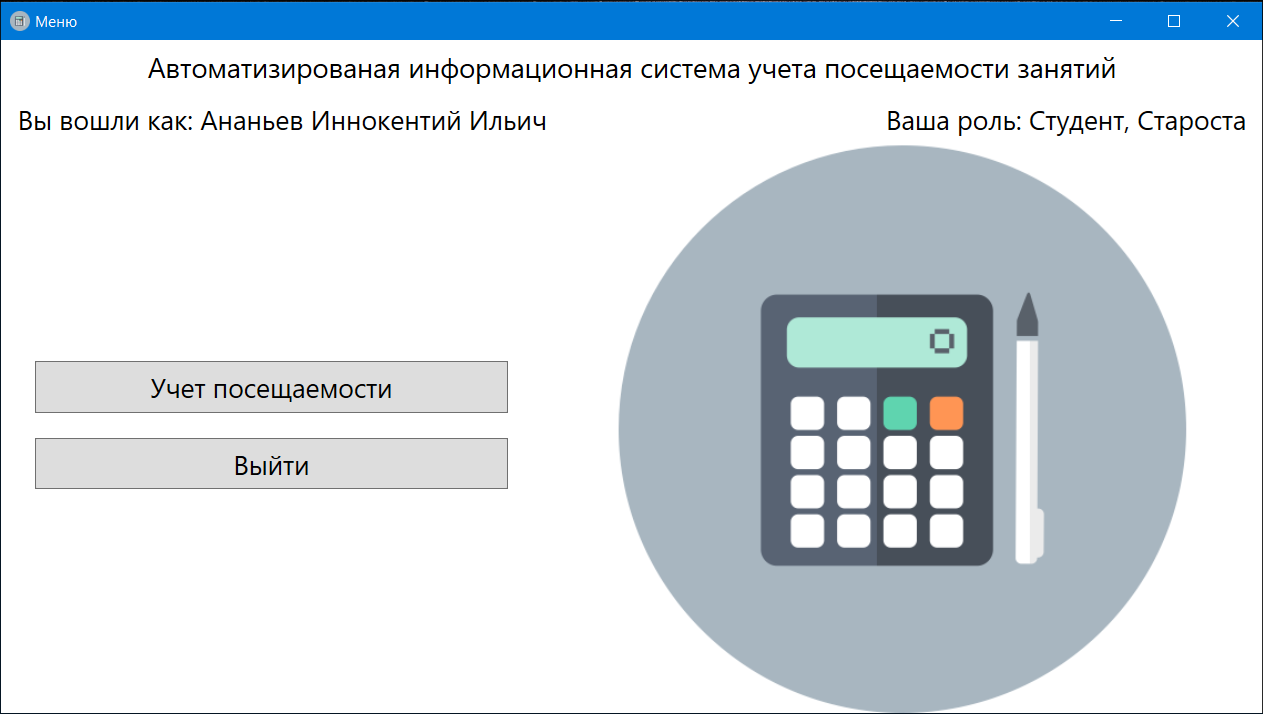


Рисунок 2.8 – Форма меню (старосты)

На рисунке 2.9 показана форма просмотра учетных записей

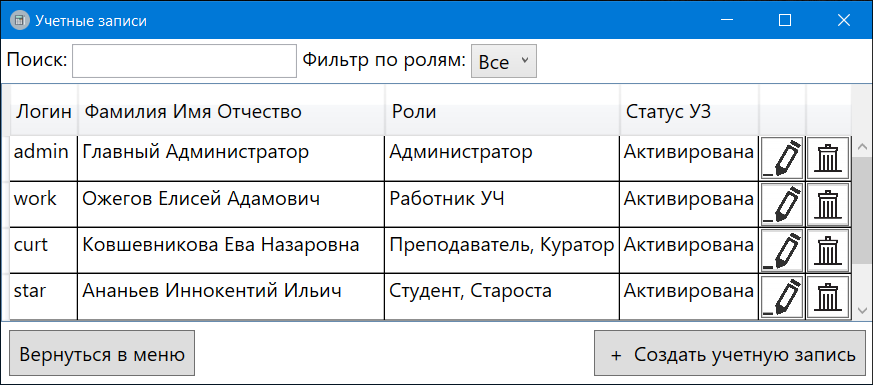


Рисунок 2.9 – Форма учетные записи

На рисунке 2.10 показана форма создания учетной записи

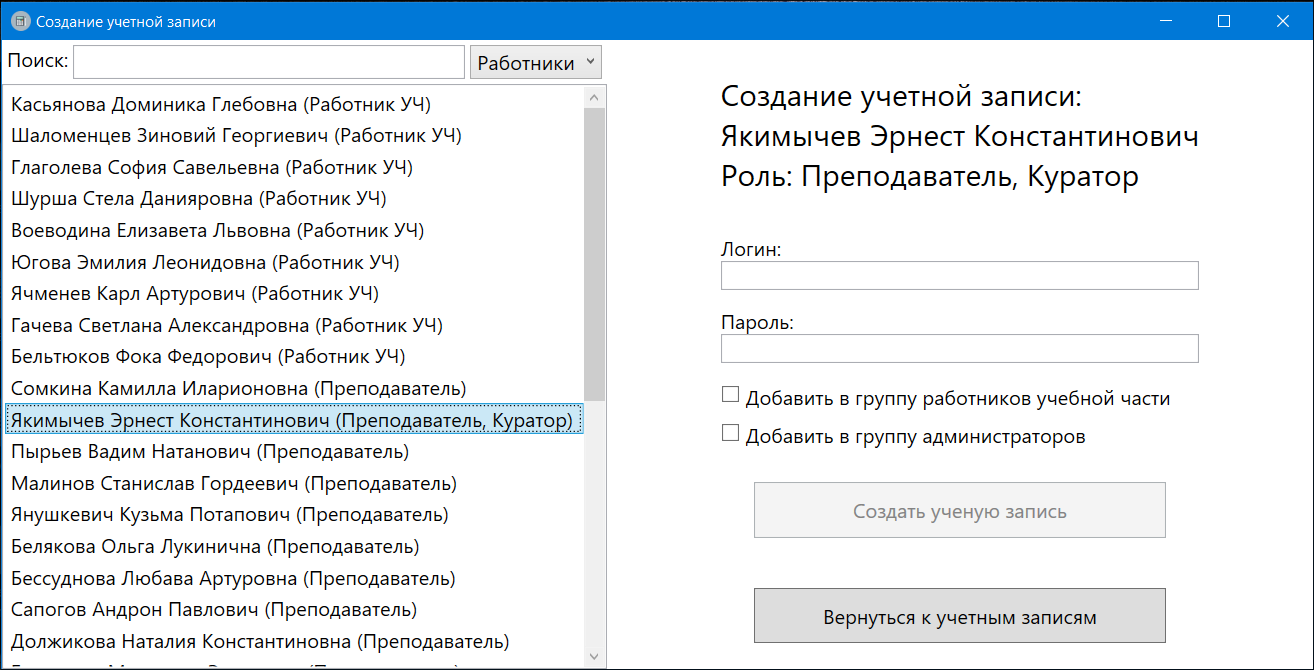


Рисунок 2.10 – Форма создание учетной записи

На рисунке 2.11 показана форма редактирования учетной записи

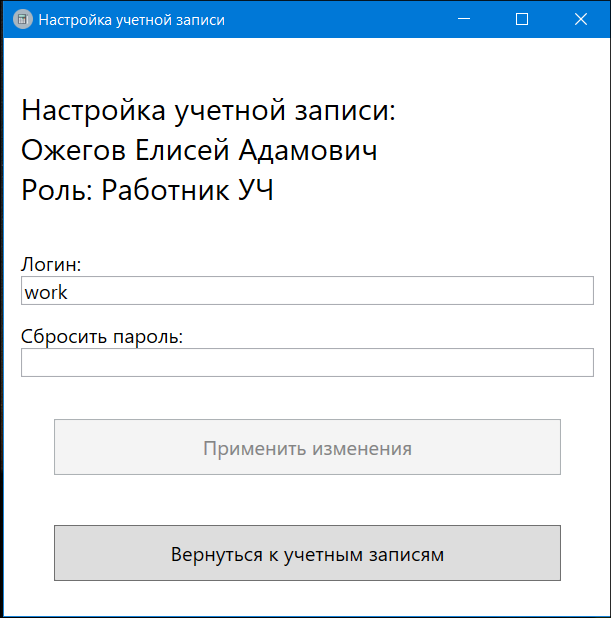


Рисунок 2.11 – Форма настройка учетной записи

На рисунке 2.12 показана форма массового импорта и экспорта данных

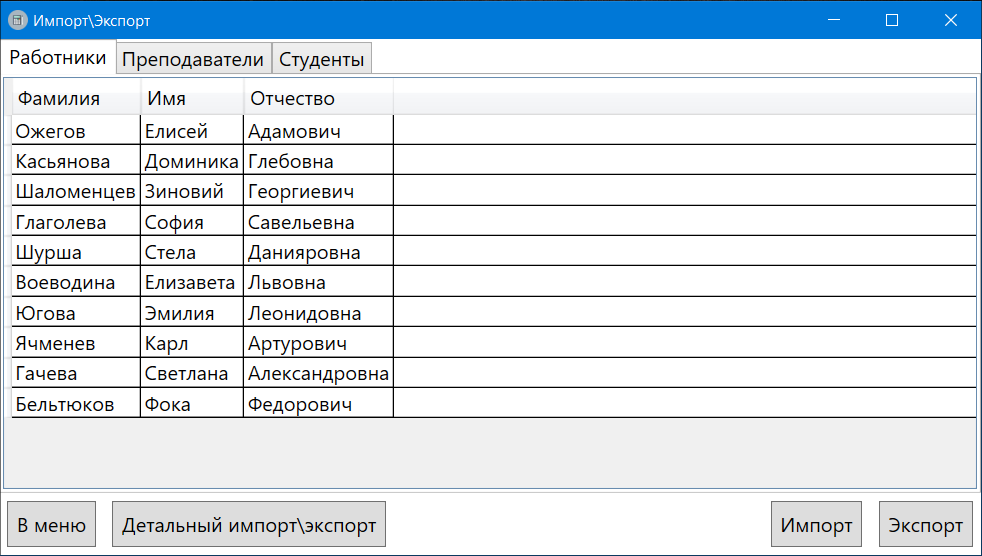


Рисунок 2.12 – Форма импорт\экспорт

На рисунке 2.13 показана форма детального импорта и экспорта данных

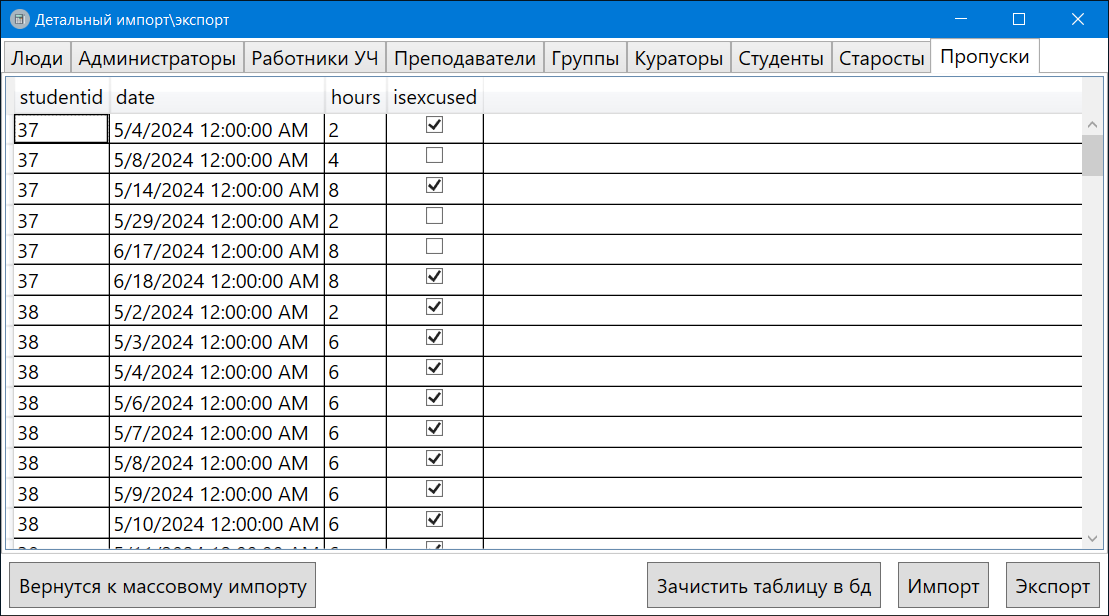


Рисунок 2.13 – Форма детальный импорт\экспорт

На рисунке 2.14 показана форма резервного копирования

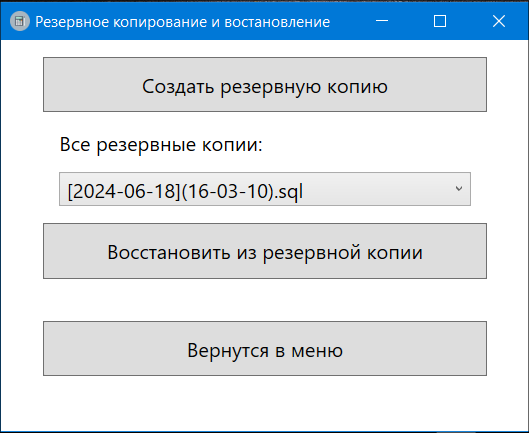


Рисунок 2.14 – Форма резервное копирование и восстановление

На рисунке 2.15 показана форма назначения кураторов

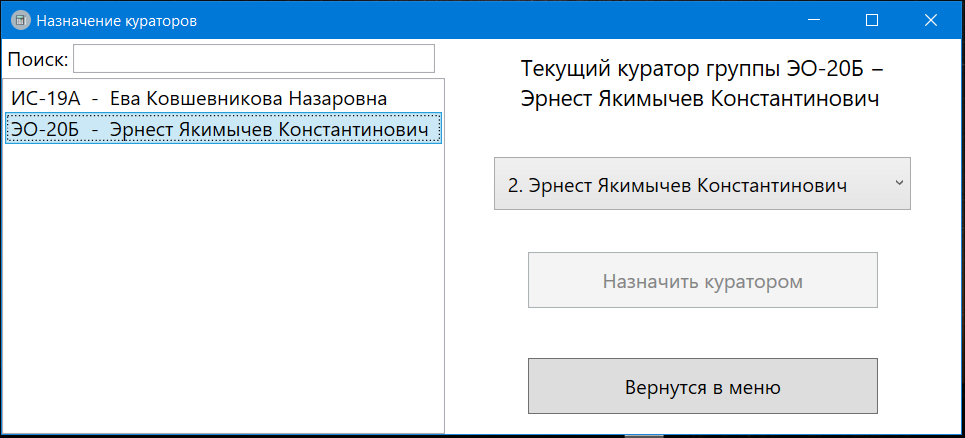


Рисунок 2.15 – Форма назначение кураторов

На рисунке 2.16 показана форма назначения старосты

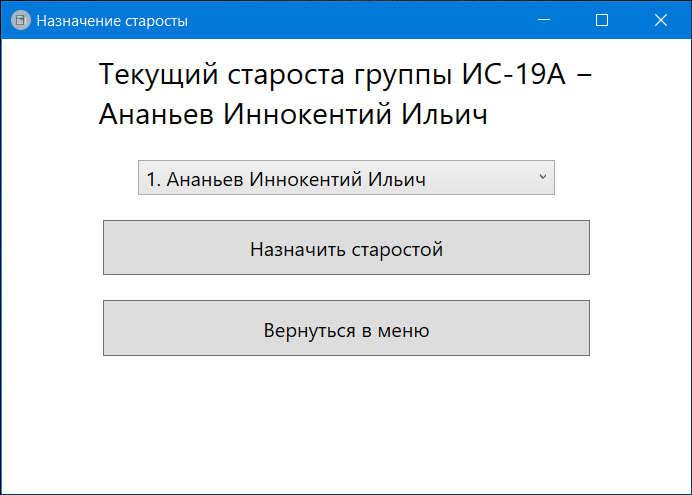


Рисунок 2.16 – Форма назначение старосты

На рисунке 2.17 показана форма просмотра посещаемости за месяц

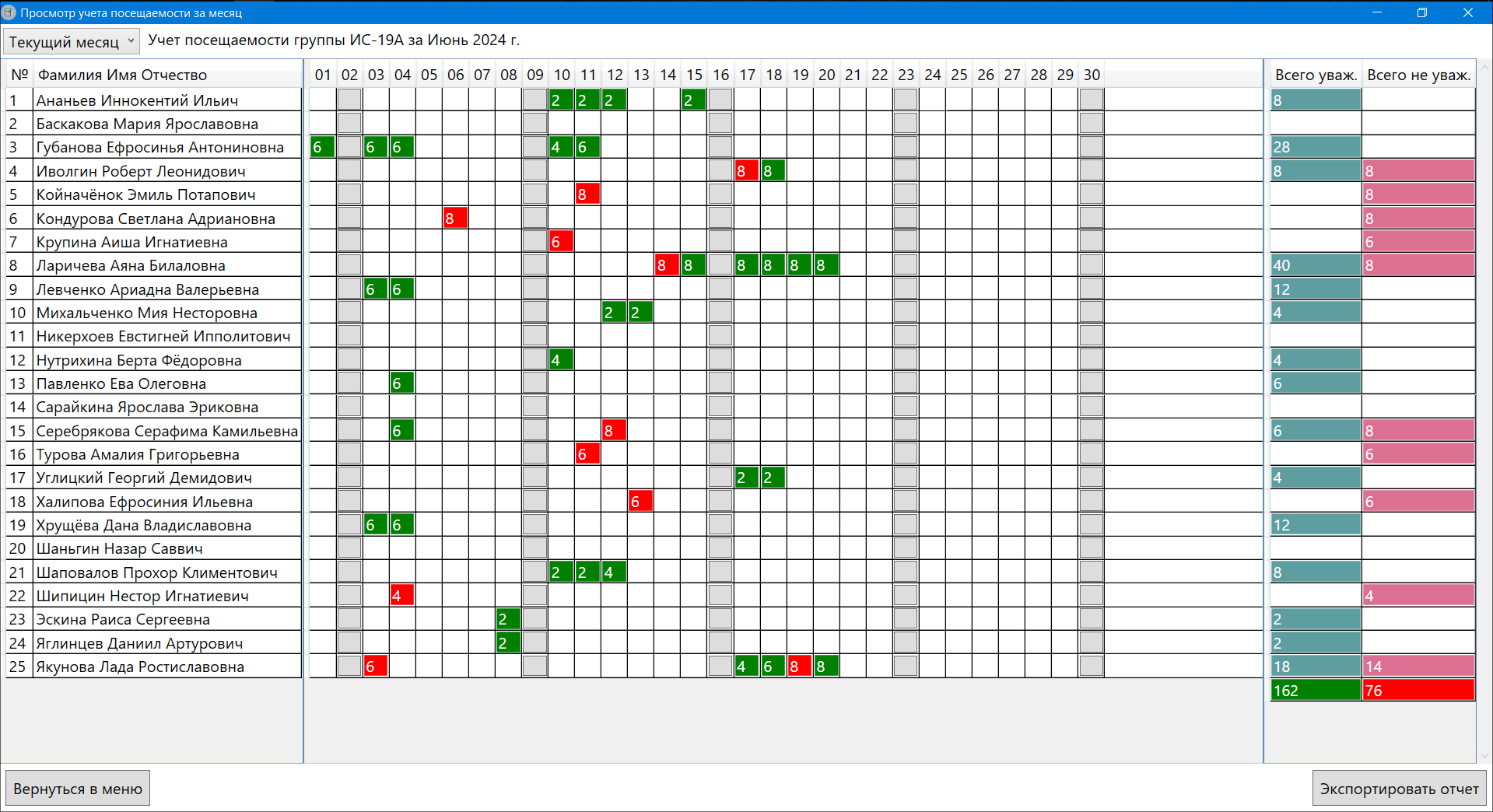


Рисунок 2.17 – Форма просмотр учета посещаемости за месяц

На рисунке 2.18 показана форма просмотра посещаемости по месяцам

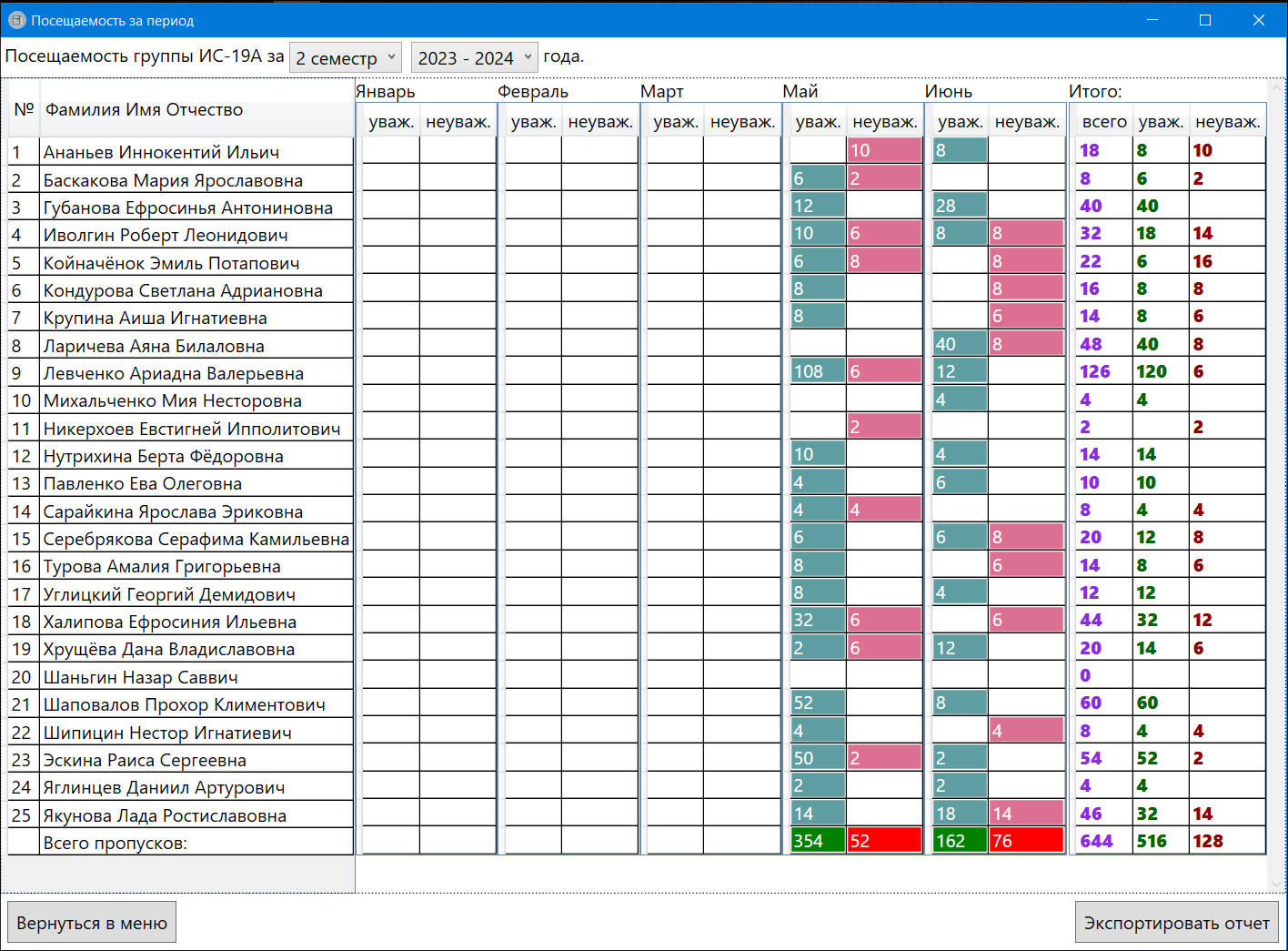


Рисунок 2.18 – Форма посещаемость за период

На рисунке 2.19 показана форма учета посещаемости за месяц

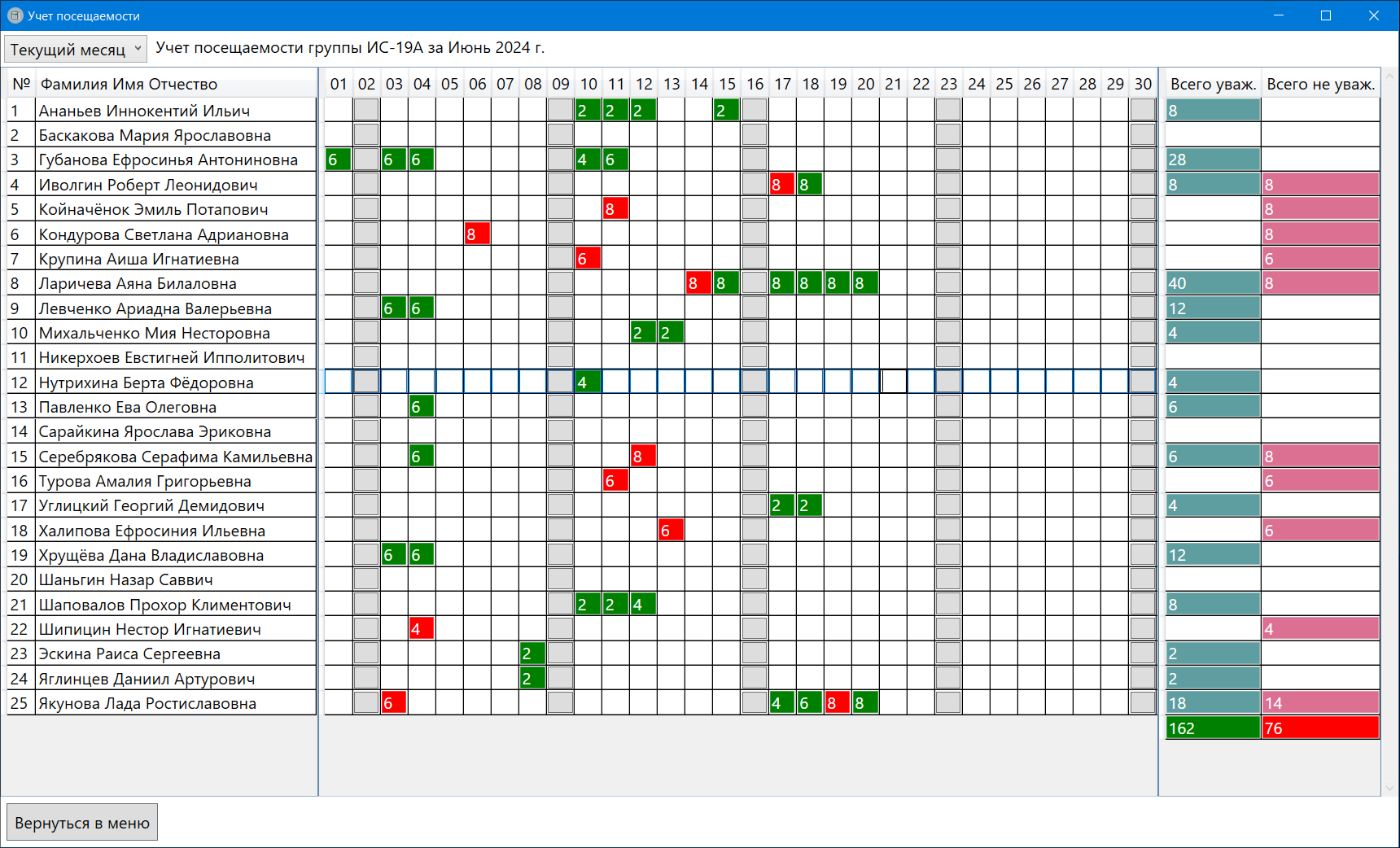


Рисунок 2.19 – Форма учет посещаемости

2.4.2 Описание основных программных модулей

Private void button2\_Click(object sender, EventArgs e)– резервное копирование;

private void button3\_Click(object sender, EventArgs e)– восстановление БД;

private void button2\_Click(object sender, EventArgs e)– экспорт в CSV;

private void button1\_Click(object sender, EventArgs e)– импорт в CSV;

private void LoadData() – вывод базы данных;

private void button1\_Click(object sender, EventArgs e) – назначение куратора

private void button2\_Click(object sender, EventArgs e) – назначение старосты

private void button3\_Click(object sender, EventArgs e) – проставление пропуска;

private void dataGridView1\_CellClick(object sender, DataGridViewCellEventArgs e) – загрузка данных при выборе месяца;

private void button3\_Click(object sender, EventArgs e) – новой учетной записи

private void button5\_Click(object sender, EventArgs e)– редактирование учетной записи;

private void button4\_Click(object sender, EventArgs e) – удаление учетной записи;

private void button1\_Click(object sender, EventArgs e) – экспорт отчета посещаемсоти за месяц;

private void button2\_Click(object sender, EventArgs e) – экспорт отчета посещаемсоти за период по месяцам;

private void CheckRole() – блокировка элементов в зависимости от роли пользователя;

private void button2\_Click(object sender, EventArgs e) – вход в систему;

public class customDataManager – сортировка, поиск и фильтрация;

private void dataGridView1\_CellFormatting(object sender, DataGridViewCellFormattingEventArgs e) – условное форматирование таблицы;

private void GetSumm() – вывод количество прогулов

private void checkPagination(bool next) – пагинация;

private void textBox1\_TextChanged(object sender, EventArgs e) – поиск данные по таблице;

private void button8\_Click(object sender, EventArgs e) – сброс фильтров поиска и сортировки;

static class excel – вывод отчета в Excel, импорт и экспорт таблиц в csv;

public static string HashPassword(string password) – хэширование пароля;

private void CreateCaptcha() – создание капчи;

private void button1\_Click(object sender, EventArgs e) – вход в учетную запись;

private void button2\_Click(object sender, EventArgs e) – резервное копирование и выход из программы;

public class DataContext – класс для работы с базой данных;

static class Backuper – класс расширения для работы с резервным копированием и восстановлением;

class AccountService – класс для работы с учетными записями;

class AttendensService – класс для работы с пропусками;

class GroupService – класс для работы с группами студентами и преподавателями;

class HashService – класс для хэширования пароля;

class ImportExportService – класс для работы с импортом и экспортом;

class RoleService – класс для работы с ролями пользователей;

public class SearchService – класс живого, умного поиска.

2.5 Применяемые меры защиты информации

Защита данных – это предупреждение несанкцианированного(случайного или умышленного) доступа к данным, их изменения или разрушения данных при сбоях аппаратных или программных средств и ошибках в работе сотрудников группы эксплуатации.

Функции:

* обеспечение безопасности данных;
* обеспечение секретности данных.

В процессе разработки информационной системы были реализованы следующие меры защиты информации:

*1. парольная защита базы данных.*

В соответствии с рисунком 2.20, парольная защита необходима для обеспечения ее защиты от несанкционированного доступа.

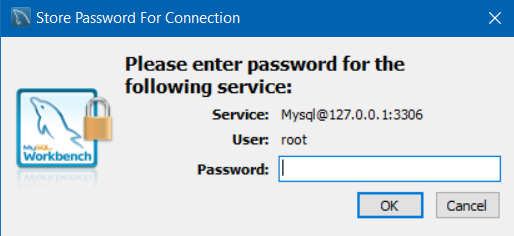


Рисунок 2.20 - Парольная защита базы данных

*2. установки прав доступа,* для того чтобы разграничить доступ пользователям к ресурсам базы. Для этого необходимо поддерживать два фундаментальных принципа: проверку полномочий и проверку подлинности (аутентификацию). Добавлены дополнительные степени защиты: наличие capcha при третьем неуданом входе и блокировка на 15 секунд при повторной неудачной попытке, в соответствии с рисунками 2.21 – 2.23

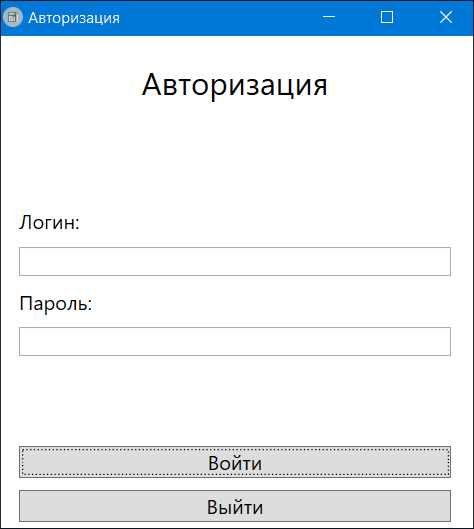


Рисунок 2.21 – Парольная защита программы

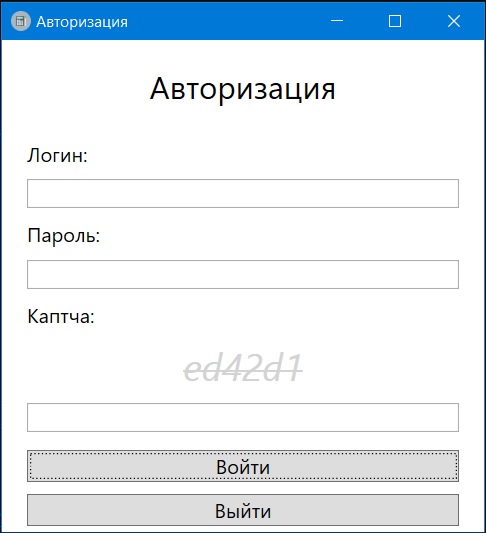


Рисунок 2.22 – капча, при 3 неверном входе

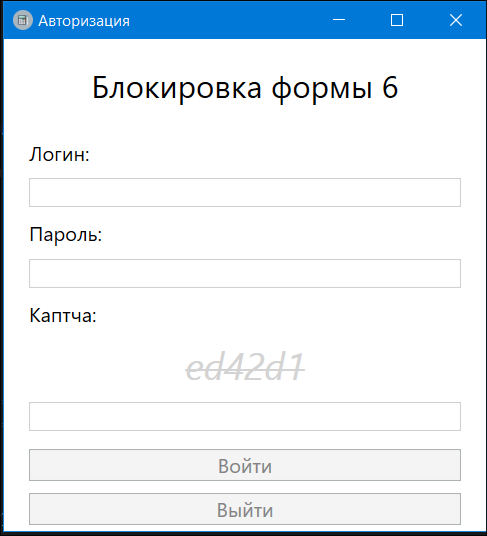


Рисунок 2.23 – блокировка формы на 15 секунд при неверном вводе капчи

Проверка полномочий основана на том, что каждому пользователю или процессу информационной системы соответствует набор действий, которые он может выполнять по отношению к определенным объектам. Проверка подлинности означает достоверное подтверждение того, что пользователь или процесс, пытающийся выполнить санкционированное действие, действительно тот, за кого он себя выдает.

*3. обязательная смена пароля* – необходимо для полной секретности паролей пользователей. Запрос на смену пароля появляется после того как пользователь вошел в систему под паролем который задал администратор, то есть либо после создания учетной записи, либо после сброса пароля администратором.

Форма запроса смены пароля представлена в соответствии с рисунком 2.24

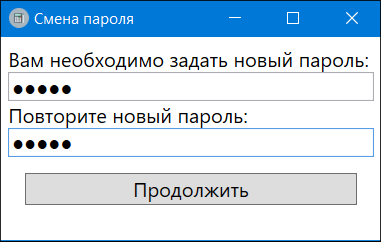


Рисунок 2.32 – Форма «Специальные возможности»

*4. резервное копирование БД -* необходимо для зашиты базы данных от системных сбоев, ошибок и для возможности восстановления продолжения работы с базой данных. Резервное копирование осуществляется автоматически при выходе из программы.

Согласно рисунку 2.32 резервное копирование и восстановление базы данных осуществялется с помощью списка резервных копий на форме Резервное копировние и восстановление.

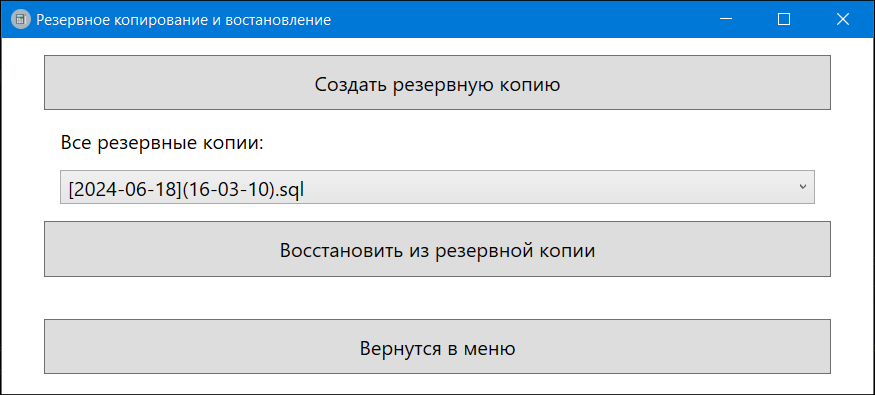


Рисунок 2.32 – Форма «Резервное копировние и восстановление»

Существует несколько причин, при которых база данных может оказаться поврежденной.

Восстановление после сбоев осуществляется при нажатии кнопки «Восстановить базу», перед этим нужно выбрать из спика копию, которую нужно восстановить.

**3 Тестирование**

Тестирование – ряд мероприятий, связанных с различного рода испытаниями объекта тестирования с целью установления соответствия или несоответствия его характеристик определенным требованиям и выявления дефектов. Дефектами, в свою очередь, могут быть как ошибки в работе, так и неприемлемое качество функционирования в определенных условиях эксплуатации.

Результаты тестирования- это список выявленных несоответствий и дефектов, но, как правило, без указания их причин.

Тестирование программного продукта проводят как на этапах его создания, так и на этапах технического сопровождения и дальнейшего развития. Оно является одним из наиболее устоявшихся способов обеспечения качества разработки программного обеспечения и входит в набор эффективных средств современной системы обеспечения качества программного продукта.

Таблица 3.1 – Тестирование программы

|  |  |  |
| --- | --- | --- |
| Название элемента формы | Ожидаемый результат | Полученный результат |
| 1 | 2 | 3 |
| **Форма авторизации** | | |
| Зволнение поля логин | Произвольный ввод | Корректное выполнение команды |
| Зволнение поля пароль | Произвольный скрытый ввод | Корректное выполнение команды |
| Кнопка «Войти» | При правильно веденных логине и пароле – переход в главное меню, соответственно роли пользоватля | Корректное выполнение команды |
| Кнопка «Войти» | При ошибке логина или пароля 3 раза появление капчи | Корректное выполнение команды |
| Кнопка «Ответить» | При ошибке капчи блокировка на 15 секунд и смена текста на капче | Корректное выполнение команды |
| Кнопка «Выйти» | Завершение работы приложения и создание резервной копии базы | Корректное выполнение команды |

Продолжение таблицы 3.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | | 3 | |
| **Форма главного меню администратора** | | | | |
| Кнопка «Учетные записи» | | Переход на форму просмотра учетных записей | | Корректное выполнение команды |
| Кнопка «Импорт\Экспорт» | | Переход на форму массового импорта и экспорта | | Корректное выполнение команды |
| Кнопка «Востановление данных» | | Переход на форму массового импорта и экспорта | | Корректное выполнение команды |
| Кнопка «Выйти» | | Выход из аккаунта с переходом на форму авторизации | | Корректное выполнение команды |
| **Форма главного меню работника учебной части** | | | | |
| Кнопка «Назначение кураторов» | | Переход на форму назначения кураторов | | Корректное выполнение команды |
| Кнопка «Выйти» | | Выход из аккаунта с переходом на форму авторизации | | Корректное выполнение команды |
| **Форма главного меню куратора** | | | | |
| Кнопка «Посещаемость за месяц» | | Переход на форму просмотра посещаемости за месяц | | Корректное выполнение команды |
| Кнопка «Посещаемость за период» | | Переход на форму просмотра посещаемости за период по месяцам | | Корректное выполнение команды |
| Кнопка «Назначение старосты» | | Переход на форму назначения старосты | | Корректное выполнение команды |
| Кнопка «Выйти» | | Выход из аккаунта с переходом на форму авторизации | | Корректное выполнение команды |
| **Форма главного меню старосты** | | | | |
| Кнопка «Учет посещаемости» | | Переход на форму добавления продукции | | Корректное выполнение команды |
| Кнопка «Выйти» | | Выход из аккаунта с переходом на форму авторизации | | Корректное выполнение команды |

Продолжение таблицы 3.1

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| **Форма просмотра учетных записей** | | |
| Ввод в поле «Поиск» | В таблице учетных записей подходящие поднимаются вверх | Корректное выполнение команды |
| Смена фильтра «Фильтр по ролям» | В таблице учетных записей фильтрация в соответсвии со значением | Корректное выполнение команды |
| Кнопка «Удалить» у любой учетной записи | Удаление учетной записи | Корректное выполнение команды |
| Кнопка «Редактировать» у любой учетной записи | Открытие формы настройки учетной записи | Корректное выполнение команды |
| Кнопка «Создать учетную запись» | Открытие формы создания учетной записи | Корректное выполнение команды |
| Кнопка «Вернутся в меню» | Возврат на форму «Меню» | Корректное выполнение команды |
| **Форма создание учетной записи** | | |
| Ввод в поле «Поиск» | В списке людей подходящие поднимаются вверх | Корректное выполнение команды |
| Смена сортировки на «Работники» | Фильтрация списка по работникам, появление чекбоксов добавления в группы администраторов и работников учебной части | Корректное выполнение команды |
| Смена сортировки на «Студенты» | Фильтрация списка только по студентам, скрытие чекбоксов добавления в группы | Корректное выполнение команды |
| Выбор человека из списка | Заполнение ФИО и ролей | Корректное выполнение команды |
| Ввод в поле «Логин» | При вводе более менее 4 букв или цифр – блокировка кнопки создать учетну запись | Корректное выполнение команды |
| Ввод в поле «Пароль» | При вводе более менее 4 букв или цифр – блокировка кнопки создать учетну запись | Корректное выполнение команды |
| Кнопка «Создание учетной записи» | Создание учетной записи, возврат на форму «Просмотр учетных записей», и подсвечивание новой учетной записи | Корректное выполнение команды |
| Кнопка «Вернутся к учетным записям» | Возврат на форму «Просмотр учетных записей», | Корректное выполнение команды |
| **Форма настройка учетной записи** | | |
| Изменение поля «Логин» | При вводе более менее 4 букв или цифр – блокировка кнопки создать учетну запись, иначе - разблокировка | Корректное выполнение команды |
| Поля «Пароль» | При вводе более менее 4 букв или цифр – блокировка кнопки создать учетну запись, иначе разблокировка | Корректное выполнение команды |
| Кнопка «Применить изменения» | Редактирование учетной записи | Корректное выполнение команды |
| Кнопка «Вернутся к учетным записям» | Возврат на форму «Просмотр учетных записей», | Корректное выполнение команды |
| **Форма массового импорта и экспорта** | | |
| Выбор вкладки | Изменение текущей таблицы, отображение | Корректное выполнение команды |
| Кнопка «Импорт» | Импорт текущей таблицы | Корректное выполнение команды |
| Кнопка «Экспорт» | Экспорт текущей таблицы | Корректное выполнение команды |
| Кнопка «Детальный импорт\экспорт» | Переход на форму детального импорта иэкспорта | Корректное выполнение команды |
| Кнопка «В меню» | Переход на форму главного меню | Корректное выполнение команды |
| **Форма детального импорта и экспорта** | | |
| Выбор вкладки | Изменение текущей таблицы, отображение | Корректное выполнение команды |
| Кнопка «Импорт» | Импорт текущей таблицы | Корректное выполнение команды |
| Кнопка «Экспорт» | Экспорт текущей таблицы | Корректное выполнение команды |
| Кнопка «Зачистить таблицу» | Удаление всех записей и с ними связаных в бд | Корректное выполнение команды |
| Кнопка «Вернутся к массовому импорту» | Переход на форму массового импорта | Корректное выполнение команды |
| **Форма резервного копирования и восстановления** | | |
| Кнопка «Создать резервную копию» | Создание резервной копии | Корректное выполнение команды |
| Выбор резервной копии | Свободный выбор | Корректное выполнение команды |
| Кнопка «Восстановить из резервной копии» | Востановление из выбраной резервной копии | Корректное выполнение команды |
| Кнопка «Вернутся в меню» | Переход на форму главного меню | Корректное выполнение команды |
| **Форма назначение кураторов** | | |
| Ввод в поле «Поиск» | В списке групп подходящие поднимаются вверх | Корректное выполнение команды |
| Выбор группы из списка | Заполнение группы и ФИО текущего куратора | Корректное выполнение команды |
| Выбор преподавателя из списка | Если преподаватель еще не куратор, то разболкировка кнопки назначить куратором | Корректное выполнение команды |
| Кнопка «Назначить куратором» | Назначает куратором выбраной группы | Корректное выполнение команды |
| Кнопка «Вернутся в меню» | Переход на форму главного меню | Корректное выполнение команды |
| **Форма просмотр учета посещаемости за месяц** | | |
| Кнопка «Текущий\Предидущий месяц» | Переключение между текущим и предидущим месяцем | Корректное выполнение команды |
| Кнопка «Экспортировать отчет» | Экспорт отчета Excel | Корректное выполнение команды |
| Кнопка «Вернутся в меню» | Переход на форму главного меню | Корректное выполнение команды |
| **Форма просмотр учета посещаемости за период** | | |
| Выбор семестра | Отображение данных только за выбраный семестр | Корректное выполнение команды |
| Выбор учебного года | Отображение данных только за выбраный год | Корректное выполнение команды |
| Кнопка «Экспортировать отчет» | Экспорт отчета Excel | Корректное выполнение команды |
| Кнопка «Вернутся в меню» | Переход на форму главного меню | Корректное выполнение команды |
| **Форма назначения старосты** | | |
| Выбор студента | Свободный выбор | Корректное выполнение команды |
| Кнопка назначить старостой | Назначение нового старосты, заполнение полей | Корректное выполнение команды |
| Кнопка «Вернутся в меню» | Переход на форму главного меню | Корректное выполнение команды |
| **Форма учета посещаемости за месяц** | | |
| Кнопка «Текущий\Предидущий месяц» | Переключение между текущим и предидущим месяцем | Корректное выполнение команды |
| Ввод в клетку пропуска | Ввод числа от 1 до 8, создание не уважительного пропуска | Корректное выполнение команды |
| ЛКМ + ПКМ по клетке пропуска | Если пропуск уважителен – появится кнопка пометить как неуважительный,  Если пропуск не уважителен – появится кнопка пометить как уважительный,  Если пропуска нет, то ничего. | Корректное выполнение команды |
| Кнопка «Пометить как неуважительный» | Пропуск помечается как неуважительный | Корректное выполнение команды |
| Кнопка «Пометить как уважительный» | Пропуск помечается как уважительный | Корректное выполнение команды |
| Кнопка «Вернутся в меню» | Переход на форму главного меню | Корректное выполнение команды |

**4 Экономическое обоснование**

Разрабатываемое программное приложение предназначено для организации учета посещаемости занятий. Существующие программные аналоги имеют обширный функционал, рассчитанный на различные задачи, такие как учет посещаемости, расписаний, регистрации студентов, управление статусами посещаемости, формирование отчетов и статистики. Все программные продукты платные, требуют предварительной настройки и сопровождения, которые являются платными услугами. Аналоги диктуют четкие правила принципов работы системы.

В рамках выпускной квалификационной работы было выбрано одно направление – «Учет посещаемости занятий». Реализация данного направления достаточна для демонстрации профессиональных навыков программирования в области автоматизации деятельности преподавателей, администраторов и учебных заведений на платформе .NET.

Разрабатываемая АИС позволит автоматизировать процессы:

- занесение пропусков занятий, уважительных и нет;

- учет посещаемости с использованием поиска, сортировки, условного форматирования и различных фильтров;

- управлять ролями пользователей;

- формирование отчета посещаемости за определенный период с подсчетом и экспортировать документы в приложения MS Excel 2007

- выполнять импорт и экспорт данных, резервное копирование и восстановление БД.

Программное приложение позволит:

- оперативно осуществлять выборку данных из базы;

- редактировать базу данных.

Стадии разработки программного продукта представлены в таблице 4.1

Таблица 4.1 – Стадии и этапы разработки

| № п/п | Название этапа | Содержание | Кол-во дней |
| --- | --- | --- | --- |
| 1 | Анализ | Анализ предметной области | 2 |
| 2 | Проектирование | Составление технического задания | 3 |
| 3 | Разработка | Разработка структуры БД | 26 |
| Разработка форм программного приложения |
| Разработка программного кода |
| 4 | Тестирование | Проверка работоспособности всех функций программного продукта | 1 |
| 5 | Документирование | Составление руководства пользователя | 3 |
| Составление листинга | 1 |
| Составление пояснительной записки | 5 |

На изготовление программного продукта будет потрачен 40 дней.

Приведенные в данном разделе выпускной квалификационной дипломной работы результирующие таблицы, перечни данных, позволяют сопоставить результаты разработки и затраты на нее, чтобы сделать вывод об эффективности проекта. Исходные данные для расчета экономических показателей приведены в таблице 4.2.

Таблица 4.2 Исходные данные для расчета экономических показателей

|  |  |  |  |
| --- | --- | --- | --- |
| Обозначение | Наименование показателя | Единицы измерения | Значение показателя |
| СЭВМ | Стоимость ЭВМ | руб. | 55000 |
| ДМ | Среднее количество рабочих дней в месяце | дни | 24 |
| Счас | Стоимость час работы программиста | руб. | 300 |
| ЦЭЛ | Тариф за 1 кВт/час | руб. | 4,48 |
| Тчас | Количество часов работы над разработкой в день | час | 5 |
| P | Мощность, потребляемая ЭВМ | кВт | 0,8 |
| Тинтер | Тариф за услугу Интернет, месяц | руб. | 1000 |

4.1 Расчет себестоимости программного продукта

Себестоимость разработки программного продукта рассчитываются по следующей формуле:

|  |  |
| --- | --- |
| Спп = ЗП + Зэл + А + Зпр | (1) |

где ЗП– затрата программиста;

Зэл – затраты на электроэнергию;

А – амортизация оборудования;

Зпр – прочие расходы.

Зарплата программиста рассчитывается по формуле:

|  |  |
| --- | --- |
| ЗП **=** Сч **×** Тчас **×** Д | (2) |

где Сч – стоимость час работы программиста (руб.);

Тчас – количество часов работы над разработкой в день;

Д– количество дней затраченных на разработку программного продукта.

Согласно таблице 4.1 на разработку программного продукта ушло 24 рабочих дня. Среднее количество часов работы в день составило 5 часа. Так как навыков разработки программных продуктов мало, то часовая ставка начинающего программиста составит 300 рублей, что соответствует данным таблицы 4.2

|  |  |
| --- | --- |
| ЗП **=** 300 **×** 5**×** 24 = 36000 руб. |  |

Затраты на электроэнергию рассчитывается по формуле:

|  |  |
| --- | --- |
| Зэл **=** Чотраб **×** Ц1кВт **×** Р | (3) |

где Чотраб – количество часов работы оборудования (ПК), освещения;

Ц1кВт – стоимость 1кВт электроэнергии, (руб.);

Р – мощность, потребляемая ЭВМ, (кВт).

Количество часов работы оборудования (ПК), освещения рассчитывается по формуле:

|  |  |
| --- | --- |
| Чотраб **=** Тчас **×** Д | (4) |

где Тчас – количество часов работы над разработкой в день;

Д– количество дней затраченных на разработку программного продукта.

Количество дней на разработку составило 24, что соответствует данным таблицы 4.1. Количество часов работы в день 5, в соответствии с таблицей 4.2

|  |  |
| --- | --- |
| Чотраб **=** 5**×** 24 = 120 часов |  |

В соответствии с данными таблицы 4.2 стоимости 1кВт электроэнергии равна 4,48 рублей, а потребляемая мощность ПК составляет 0,65 кВт в час.

|  |  |
| --- | --- |
| Зэл **=** 120 **×** 4,48 **×** 0,8= 430,08 руб. |  |

Амортизационные отчисления рассчитываются с учетом нормы амортизации (Нам **=** 5-10%):

|  |  |
| --- | --- |
| А **=** Нам **×** СЭВМ | (5) |

где СЭВМ – стоимость оборудования, согласно таблице 4.2.

|  |  |
| --- | --- |
| А **=** 0,05 **×** 55000 = 2750 руб. |  |

Для разработки программного продукта необходимо использовать Интернет, следовательно определим затраты на Интернет, как прочие.

|  |  |
| --- | --- |
| З пр **=** Ц инт (1ч) **×** Тчас **×** Д | (6) |

Цена одного часа работы в Интернет рассчитывается по формуле:

|  |  |
| --- | --- |
| Ц инт (1ч) = Цтар : Дмес : 24 | (7) |

Согласно таблице 4.2 стоимость услуги Интернет в месяц составил 600 рублей. Исходя из стоимости тарифа определим стоимость часа работ в Интернет.

|  |  |
| --- | --- |
| Ц инт (1ч) = 1000 : 24 : 24 = 1,74 руб.  З пр **=** 1,74 **×** 5**×** 24 = 280,8 руб. |  |

Выполнив все расчеты, определим себестоимость программного продукта.

|  |  |
| --- | --- |
| Спп = 36000 + 430,08 + 2750 + 280,8 = 39388,88 рублей ≈ 40000 рублей |  |

4.2 Стратегия продвижения программного продукта на рынок

Сегодня самым эффективным способом продажи программного продукта является Интернет.

Наиболее распространёнными средствами реализации программного продукта в сети Интернет являются: демоверсия, реклама, размещение на различных хостингах.

- демоверсия продукта — предварительная (ограниченная и, как правило, бесплатная) версия продукта. Демоверсии дают представление пользователям о готовящемся продукте и создаются в качестве рекламы и привлечения вниманию к будущему продукту. В этом смысле они приравниваются к трейлерам для кинофильмов. Обычно они являются в некотором смысле не готовой к продаже версией полного продукта. По этим причинам в большинстве случаев они распространяются бесплатно;

- реклама (баннеры) - один из преобладающих форматов [Интернет-рекламы](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82-%D1%80%D0%B5%D0%BA%D0%BB%D0%B0%D0%BC%D0%B0). Представляет собой графическое изображение, аналогичное рекламному модулю в прессе, но способное содержать анимированные (редко видео-) элементы, а также может являться [гиперссылкой](https://ru.wikipedia.org/wiki/%D0%93%D0%B8%D0%BF%D0%B5%D1%80%D1%81%D1%81%D1%8B%D0%BB%D0%BA%D0%B0) на сайт рекламодателя или страницу с дополнительной информацией;

- размещение на специализированных хостингах – также вариантом распространения программного продукта является размещение на специализированных хостингах, которые являются своего рода специализированными магазинами программного обеспечения.

Из выше перечисленных вариантов распространения программного продукта, было выбрано размещение дипломного проекта на сайте CodeCanyon.net, который взимает плату 30% от стоимости работы. Дипломные работы вирируются в размере от 3000 до 16000 рублей.

4.3 Расчет цены разработанной программы

Цена разработанной программы определяется по следующей формуле:

|  |  |
| --- | --- |
| Ц пп = Спп | (8) |

где Спп – себестоимость программного продукта (руб.);

Зпродв – суммарные затраты на продвижение программы (руб.);

Ц пп = 40000 руб

Выручка от продаж при условии  – количество пользователей, желающих прибрести программу, составит:

|  |  |
| --- | --- |
|  | (10) |

Было принято решение продавать информационную систему по ведению базы данных за 5000 рублей. При продаже каждой копии выручка составит 4000 рублей.

Срок окупаемости инвестиционного проекта () – это период времени, который потребуется для возмещения инвестиций. – определяется с учетом дисконтирования, путем суммирования ежегодных поступлений до определенного периода, в котором они превзойдут первоначальные расходы денежных средств. Расчёт срока окупаемости представлен в таблице 4.3

Таблица 4.3 Расчет срока окупаемости разработанной программы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Период | Цена ПП,  руб. | Кол-во покупок  шт. | Выручка,  руб. | Накапливаемая выручка,  руб. |
| 1 год | 5000 | 6 | 24000 | 24000 |
| 2 год | 5000 | 4 | 16000 | 40000 |
| Итого | 5000 | 10 | 14700 |  |

4.4 Технико-экономические показатели разработки программы

Обобщенные технико-экономические показатели разработки программы сведены в таблицу 4.4.

Таблица 4.4 – Технико-экономические показатели разработки программы

|  |  |
| --- | --- |
| Показатель | Значение |
| Себестоимость программного продукта (руб.) | 40000 |
| Цена разработки ПП (руб.) | 40000 |
| Отпускная цена (руб.) | 5000 |
| Срок окупаемости (года) | 2 |

Данные произведенные расчеты показывают, что разработанный программный продукт является эффективным с экономической точки зрения.

**Заключение**

В ходе выполнения выпускной квалификационной работы была создана автоматизированная инфомационня систем учета посещаемости занятий, отвечающее поставленным требованиям.

В процессе написния программного кода, использовался язык программирования высокого уровня. Были улучшены навыки по работе с платформой .Net, БД MYSQL, а так же приобретен навык по разработке подобных проектов.

Программное приложение обеспечивается выполнение следующих функций:

- авторизация пользователя;

- автоматизации ввода исходных данных (выпадающие списки);

- просмотр, редактирование, добавление и удаление записей в БД в режиме администратора;

- просмотр, редактирование, добавление и удаление записей при оформлении заказа, в режиме администратора;

-просмотр посещаемости, создание отчета за выбранный период в режиме куратора;

- управление ролями пользователя заказа;

- поиск, сортировка и фильтрация записей по различным критериям;

- расчет всех уважительных и неуважительных пропусков студентов;

- проверка на корректность ввода данных;

- формирование отчетов для экспорта в приложение MSExcel 2007;

- резервное копирование и восстановление БД в режиме администратора.

Программный продукт имеет архитектуру толстый клиент – сервер БД и позволяет в дальнейшем наращивать функционал переходя на другой уровень архитектуры, например разработка и подключение сайта магазина. Программа может использоваться в качестве демонстрации функционала для заключения договора с потенциальным покупателем. После внедрения возможно расширение и доработка функционала.

**Литература**

1. Эндрю Троелсен., Язык программирования C# и платформа .NET., 2015г.
2. Алексей Васильев., Программирование на C# для начинающий. Основные сведения, 2018г.
3. Троелсен Эндрю, Джепикс Филипп, Язык программирования C#7 и платформы .NET и .NET Core, 2018г.
4. Евдокимов П. В.,C# на примерах, 2019г.
5. Вагнер Билл., Наиболее эффективное программирование на C#. 50 способов улучшения кода, 2017г.
6. Мюллер Джон Поль, Семпф Билл ., C# для чайников ,2019г.