

**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ**

**Ульяновский Государственный Университет**

**Факультет Математики, Информационных и Авиационных технологий**

**Кафедра Информационных технологий**

Работа допущена к защите

Зав.кафедрой ИТ\_\_Волков М.А.\_\_\_\_

(Ф.И.О)

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

(подпись)

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

( дата )

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

-------------------------------------------------------------------------------------

(название темы)

Прикладная информатика (Бакалавриат) – 09.03.03

--------------------------------------------------------------------------------------

(наименование и номер направления)

Проект выполнил студент \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

группа подпись, дата Ф.И.О.

Научный руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

должность подпись, дата Ф.И.О.

Рецензент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

должность подпись, дата Ф.И.О.

У Л Ь Я Н О В С К

2024 г.

# Введение

**Тема исследования**

Тема данной работы посвящена разработке информационной системы для автоматизации администрирования и мониторинга учебного процесса в образовательных учреждениях, с акцентом на создание оптимизированного и доступного решения для небольших частных образовательных организаций.

**Актуальность темы**

Актуальность темы обусловлена тем, что в условиях современного образовательного процесса все большее значение приобретает автоматизация административных и мониторинговых задач. В Российской Федерации наиболее распространенным решением для автоматизации администрирования является система "1С:Предприятие". Однако, несмотря на ее популярность, на практике данное программное обеспечение часто оказывается перегруженным и медленным, что приводит к значительным временным затратам на выполнение запросов и увеличению нагрузки на преподавателей и администраторов.

Перегруженность системы 1С связана с постоянным добавлением новых функций и требований пользователей без должной оптимизации. В результате, выполнение сложных запросов к базе данных может занимать значительное время, что негативно сказывается на эффективности работы образовательных учреждений. В связи с этим возникает необходимость в разработке более оптимизированного и доступного решения, которое бы удовлетворяло потребности небольших частных образовательных организаций.

**Объект и предмет исследования**

Объектом исследования является процесс автоматизации административных и мониторинговых задач в образовательных учреждениях. Предметом исследования выступает информационная система для автоматизации администрирования и мониторинга учебного процесса.

**Цель и задачи исследования**

Целью данной работы является разработка интегрированной информационной системы для автоматизации администрирования и мониторинга учебного процесса, которая будет проста в использовании и доступна для небольших образовательных учреждений.

Для достижения данной цели в работе поставлены следующие задачи:

Изучить и систематизировать теоретические аспекты автоматизации учебного процесса и подходов к его оптимизации.

Разработать теоретическую модель информационной системы, включающую инфологическую, даталогическую и физическую модели базы данных.

Реализовать программный продукт, включающий настольное приложение на платформе C# WinForms и веб-приложение на платформе Laravel, работающие с общей базой данных MySQL.

Провести тестирование системы и оценить ее эффективность.

**Методы исследования**

Теоретической и методологической основой исследования являются труды отечественных и зарубежных специалистов в области автоматизации учебного процесса, проектирования информационных систем и управления данными. В процессе работы использовались методы объектно-ориентированного программирования и проектирования баз данных.

**Практическая значимость**

Практическая значимость работы заключается в разработке информационной системы, которая позволяет автоматизировать административные и мониторинговые задачи в образовательных учреждениях, обеспечивая тем самым удобство и эффективность работы всех участников учебного процесса. Разработанное решение ориентировано на небольшие частные образовательные организации и может быть использовано для повышения их конкурентоспособности за счет предоставления качественных и экономически эффективных образовательных услуг.

# Часть 1. Теоретическая основа автоматизации образовательного процесса и обзор существующих решений

## **1.1. Введение в проблему автоматизации учебного процесса**

Автоматизация учебного процесса является важной задачей в условиях современного образования. Системы автоматизации позволяют значительно упростить административные и мониторинговые функции, что способствует повышению эффективности работы образовательных учреждений.

**Основные цели автоматизации**

**1. Улучшение управления учебным процессом:** Автоматизация позволяет централизованно управлять всеми аспектами учебного процесса, начиная от создания расписания занятий до оценки успеваемости студентов. Это обеспечивает более прозрачное и эффективное управление ресурсами учебного заведения[1]​​.

**2. Снижение административной нагрузки:** Административный персонал образовательных учреждений часто сталкивается с большими объемами рутинных задач, таких как обработка данных о посещаемости, регистрация студентов, создание отчетов и т.д. Автоматизация этих процессов позволяет значительно снизить нагрузку на персонал, освободив их время для выполнения более важных задач[2]

**3. Повышение прозрачности и доступности данных:** Системы автоматизации обеспечивают доступ к актуальной информации в режиме реального времени. Это позволяет администраторам, преподавателям и студентам оперативно получать необходимые данные, что способствует улучшению взаимодействия между всеми участниками учебного процесса[3]

**Проблемы, связанные с отсутствием автоматизации**

**1. Ошибки и неточности:** Ручная обработка данных часто приводит к ошибкам и неточностям, что может негативно сказаться на качестве учебного процесса. Автоматизация позволяет минимизировать количество ошибок за счет использования проверенных алгоритмов и методов[4].

**2. Затраты времени:** Ручная обработка больших объемов данных занимает много времени, что снижает общую эффективность работы учебного заведения. Автоматизация позволяет значительно сократить время, необходимое для выполнения рутинных задач[5]

**3. Низкая эффективность управления:** Отсутствие автоматизации затрудняет эффективное управление учебным процессом, так как администраторы и преподаватели не имеют доступа к оперативной информации. Это может привести к задержкам в принятии решений и снижению общей эффективности работы[6].

А**ктуальность автоматизации в современном образовании**

Современные образовательные учреждения сталкиваются с новыми вызовами, такими как увеличение числа студентов, рост требований к качеству образования и необходимость соответствия международным стандартам. В этих условиях автоматизация учебного процесса становится неотъемлемой частью успешного управления образовательным учреждением.

**1. Увеличение числа студентов:** С ростом числа студентов возрастает объем данных, которые необходимо обрабатывать и анализировать. Автоматизация позволяет эффективно управлять большими объемами данных, обеспечивая высокую скорость и точность обработки[7]​.

**2. Рост требований к качеству образования:** Современные образовательные стандарты требуют от учебных заведений предоставления качественных образовательных услуг, что невозможно без использования современных технологий и систем автоматизации[8].

**3. Соответствие международным стандартам:** Для обеспечения конкурентоспособности на международном уровне образовательные учреждения должны соответствовать мировым стандартам качества. Автоматизация учебного процесса помогает достичь этого за счет использования передовых технологий и методов управления[9]​​.

**Заключение**

Введение в проблему автоматизации учебного процесса показывает, что автоматизация является необходимым инструментом для повышения эффективности работы образовательных учреждений. Она позволяет улучшить управление учебным процессом, снизить административную нагрузку и повысить прозрачность и доступность данных. В условиях современного образования, автоматизация становится ключевым фактором успешного функционирования учебных заведений, что делает ее актуальной и востребованной.

## **1.2. Существующие решения и их недостатки**

На современном рынке представлено множество решений для автоматизации учебного процесса. В этом разделе мы рассмотрим несколько популярных систем, таких как "1С:Предприятие", Moodle, Google Classroom и Canvas, а также проанализируем их преимущества и недостатки.

Система "1С:Предприятие" "1С:Предприятие" — одна из самых распространенных систем автоматизации в Российской Федерации. Она широко используется для управления различными аспектами деятельности образовательных учреждений.

**Преимущества**:

Широкий функционал: "1С:Предприятие" охватывает различные аспекты управления учебным процессом, включая бухгалтерский учет, управление персоналом, расписание занятий и успеваемость студентов.

**Гибкость**: Система легко адаптируется под специфические нужды каждого учебного заведения, что позволяет настроить её в соответствии с конкретными требованиями.

**Недостатки**:

**Перегруженность системы**: Из-за постоянного добавления новых функций и требований пользователей система становится перегруженной, что негативно сказывается на её производительности.

**Медленная работа**: Выполнение сложных запросов к базе данных может занимать значительное время, что снижает общую эффективность работ.

**Высокая стоимость**: Внедрение и обслуживание системы требуют значительных финансовых затрат, что делает её недоступной для небольших частных образовательных организаций.[10,11,12,13]

**Система Moodle**

Moodle — это бесплатная и открытая система управления обучением (Learning Management System, LMS), которая широко используется для дистанционного обучения.

**Преимущества**:

**Бесплатность и открытый код**: Moodle доступен для всех желающих и может быть модифицирован под специфические нужды учебного заведения[14].

**Широкие возможности для дистанционного обучения**: Система позволяет создавать и управлять онлайн-курсами, проводить тесты и отслеживать успеваемость студентов[15]​​.

**Недостатки**:

**Сложность настройки**: Moodle может быть сложным в настройке и администрировании, особенно для небольших организаций без достаточной ИТ-поддержки[16]​​.

**Перегруженность интерфейса**: Изобилие функций может затруднять использование системы для начинающих пользователей[17]​​.

**Система Google Classroom**

Google Classroom — это бесплатная веб-платформа для учебных заведений, которая интегрируется с другими сервисами Google, такими как Google Drive и Google Docs.

**Преимущества**:

**Интеграция с сервисами Google**: Google Classroom позволяет легко создавать и управлять учебными материалами, а также взаимодействовать со студентами через знакомый интерфейс[18]​​.

**Простота использования**: Платформа интуитивно понятна и проста в использовании, что делает её доступной для широкого круга пользователей[19]​​.

**Недостатки**:

**Ограниченные возможности кастомизации**: В отличие от Moodle, Google Classroom предоставляет ограниченные возможности для модификации и адаптации под специфические нужды[20]​​.

**Зависимость от интернета**: Для полноценного использования платформы требуется постоянное интернет-соединение, что может быть проблемой в некоторых регионах[21].

**Система Canvas**

Canvas — это коммерческая LMS, которая активно используется в образовательных учреждениях по всему миру.

**Преимущества**:

**Интуитивный интерфейс**: Canvas предлагает удобный и интуитивно понятный интерфейс, который облегчает работу преподавателей и студентов[22].

**Широкие возможности интеграции**: Платформа поддерживает интеграцию с различными сторонними сервисами и инструментами, что расширяет её функциональность[23]​​.

**Недостатки**:

**Стоимость**: Canvas является коммерческим продуктом, и его использование требует значительных финансовых затрат, что может быть неподъемным для небольших учебных заведений[24].

**Крутая кривая обучения**: Изобилие функций и возможностей может затруднять освоение системы для новых пользователей[25]​​.

**Заключение**

На рынке представлено множество систем автоматизации учебного процесса, каждая из которых имеет свои преимущества и недостатки. "1С:Предприятие" предлагает широкий функционал и гибкость, но может быть перегруженной и дорогой.

Moodle предоставляет мощные возможности для дистанционного обучения, но требует значительных усилий для настройки.

Google Classroom отличается простотой использования, но ограничен в кастомизации.

Canvas предлагает интуитивный интерфейс и широкие возможности интеграции, но является дорогим и сложным в освоении. Выбор подходящей системы зависит от специфических нужд и возможностей каждого учебного заведения.

## **1.3. Теоретические аспекты автоматизации учебного процесса**

Автоматизация учебного процесса основывается на ряде теоретических подходов и методов, которые позволяют эффективно организовать и управлять учебной деятельностью. В этом разделе рассмотрим основные теоретические аспекты, лежащие в основе систем автоматизации учебного процесса.

**Информационные системы управления (ИСУ)**

Информационные системы управления (ИСУ) представляют собой интегрированные комплексы, предназначенные для автоматизации процессов сбора, хранения, обработки и анализа информации. Применение ИСУ для автоматизации административных задач в образовательных учреждениях позволяет улучшить координацию, контроль и анализ данных. Ключевыми аспектами ИСУ являются интеграция различных модулей и возможность адаптации под специфические требования учреждения[5]​​.

**Интеграция модулей:** ИСУ объединяет различные функциональные модули, такие как управление пользователями, расписанием, успеваемостью и посещаемостью студентов. Это позволяет создать единое информационное пространство, облегчающее доступ к данным и взаимодействие между пользователями[3]​​.

**Адаптивность:** Современные ИСУ обладают высокой степенью адаптивности, что позволяет настраивать их в соответствии с потребностями конкретного учебного заведения. Это достигается за счет модульной архитектуры и возможности добавления новых функций[6]​​.

Проектирование баз данных

Проектирование баз данных является основополагающим элементом любой системы автоматизации. Оно включает создание структурированной базы данных для хранения и управления информацией о студентах, преподавателях, расписании и успеваемости. Проектирование баз данных включает в себя создание инфологической, даталогической и физической моделей, обеспечивающих целостность и доступность данных[4]​​.

**Инфологическая модель:** Инфологическая модель описывает данные и их взаимосвязи на концептуальном уровне, не зависящем от конкретной СУБД. Она включает в себя определение сущностей, атрибутов и связей между ними[1]​​.

**Даталогическая модель:** Даталогическая модель уточняет инфологическую модель, описывая данные на логическом уровне с учетом особенностей выбранной СУБД. Она включает в себя схемы таблиц, ключей и индексов[7].

**Физическая модель:** Физическая модель определяет физическое расположение данных на носителях, обеспечивая их оптимальное хранение и доступ[2]​.

Объектно-ориентированное программирование (ООП)

Объектно-ориентированное программирование (ООП) является основой для создания модульных и легко поддерживаемых приложений. Использование методов ООП позволяет повысить гибкость и расширяемость системы. Принципы ООП, такие как инкапсуляция, наследование и полиморфизм, способствуют созданию надежного и масштабируемого программного обеспечения[26]​​.

**Инкапсуляция:** Инкапсуляция позволяет скрыть внутреннюю реализацию объектов и предоставлять доступ к ним только через публичные методы, что улучшает модульность и защищенность кода[27]​​.

**Наследование:** Наследование позволяет создавать новые классы на основе существующих, что упрощает повторное использование кода и уменьшает его дублирование[28]​​.

**Полиморфизм:** Полиморфизм позволяет объектам различных классов обрабатывать сообщения одинаково, что улучшает гибкость и расширяемость системы[29].

Пользовательский интерфейс (UI)

Разработка интуитивно понятного и удобного пользовательского интерфейса (UI) является важным аспектом автоматизации. Хорошо спроектированный UI улучшает взаимодействие пользователя с системой и повышает общую удовлетворенность от использования программного обеспечения[30].

**Интуитивность:** Интуитивный интерфейс позволяет пользователям легко понимать, как взаимодействовать с системой, что снижает время на обучение и повышает эффективность работы[31]​​.

**Удобство использования:** Удобство использования достигается за счет логической организации элементов интерфейса, применения стандартных компонентов и минимизации необходимости выполнения лишних действий[32].

**Адаптивность:** Адаптивный дизайн позволяет интерфейсу корректно отображаться на устройствах с различными разрешениями и размерами экрана, что особенно важно в условиях современного многообразия устройств[33]​​.

## **1.4. Методы решения проблемы автоматизации**

Для решения проблемы автоматизации учебного процесса используются различные методы и подходы, которые включают:

Интеграция систем

Интеграция систем предполагает объединение различных функциональных модулей в единую информационную систему, что позволяет создать целостное решение для управления учебным процессом. Это включает интеграцию модулей управления пользователями, расписанием, посещаемостью и успеваемостью студентов[5].

**Преимущества интеграции**: Интеграция позволяет централизованно управлять всеми аспектами учебного процесса, обеспечивая доступ ко всей необходимой информации из одного места. Это упрощает взаимодействие между различными подразделениями образовательного учреждения и повышает общую эффективность управления[3]​​.

**Методы интеграции**: Существует несколько подходов к интеграции систем, включая использование API (Application Programming Interface), ETL (Extract, Transform, Load) процессов и шины данных (Enterprise Service Bus). Выбор метода зависит от специфических требований и архитектуры существующих систем[6]​​.

Оптимизация запросов к базе данных

Оптимизация запросов к базе данных играет ключевую роль в повышении производительности систем автоматизации. Эффективные алгоритмы и структуры данных могут значительно сократить время выполнения запросов, что особенно важно для систем, работающих с большими объемами данных[4]​​.

**Индексация данных**: Использование индексов позволяет ускорить поиск данных в таблицах базы данных. Это один из самых распространенных методов оптимизации, который обеспечивает быстрое выполнение запросов[1].

**Нормализация данных**: Нормализация помогает устранить избыточность данных и обеспечивает целостность данных. Это достигается за счет разделения данных на связанные таблицы, что упрощает их обновление и управление[7]​​.

**Распределение нагрузки**: Использование методов распределения нагрузки, таких как шардирование (sharding) и репликация, позволяет повысить производительность и отказоустойчивость системы за счет распределения данных и запросов между несколькими серверами[2]​​.

Модульное программирование

Модульное программирование предполагает разделение системы на отдельные модули, каждый из которых выполняет определенную функцию. Это позволяет легко добавлять новые функции и изменять существующие без значительных затрат времени и ресурсов[26]​​.

**Преимущества модульного подхода**: Модульный подход обеспечивает гибкость и масштабируемость системы, что особенно важно для небольших образовательных учреждений, которые могут постепенно расширять функциональность своей системы. Это также упрощает тестирование и отладку, так как изменения в одном модуле не влияют на работу других[27]​​.

**Методы модульного программирования**: Модульное программирование включает использование таких концепций, как инкапсуляция, абстракция и интерфейсы. Эти концепции позволяют разработчикам создавать независимые модули, которые взаимодействуют друг с другом через четко определенные интерфейсы[28]​​.

## **1.5. Примеры успешных внедрений**

Системы автоматизации учебного процесса демонстрируют множество примеров успешного внедрения. Рассмотрим несколько примеров использования таких систем в образовательных учреждениях и их влияние на улучшение управления учебным процессом.

Пример 1: Внедрение системы "1С:Предприятие" в крупном государственном университете

Крупный государственный университет внедрил систему "1С:Предприятие" для автоматизации своих административных процессов. Целью внедрения было улучшение управления ресурсами университета, сокращение времени на выполнение рутинных задач и повышение прозрачности данных о посещаемости и успеваемости студентов[34].

**Результаты внедрения**:

Значительное сокращение времени, необходимого для обработки данных о посещаемости и успеваемости.

Повышение точности данных за счет минимизации человеческих ошибок.

Увеличение прозрачности и доступности данных для преподавателей и администрации.

Пример 2: Использование Moodle в дистанционном обучении

Малый частный колледж использовал систему Moodle для организации дистанционного обучения. Moodle позволил создать и управлять онлайн-курсами, проводить тестирования и отслеживать успеваемость студентов, что было особенно важно в условиях пандемии COVID-19[35]​​.

**Результаты внедрения**:

Возможность продолжать образовательный процесс в условиях карантина.

Улучшение взаимодействия между студентами и преподавателями через онлайн-платформу.

Возможность для студентов учиться в удобное для них время и в своем темпе.

Пример 3: Внедрение Google Classroom в средней школе

Средняя школа внедрила Google Classroom для упрощения взаимодействия между учителями и учениками. Платформа позволила легко создавать и управлять учебными материалами, а также интегрироваться с другими сервисами Google, такими как Google Drive и Google Docs[18].

**Результаты внедрения**:

Сокращение времени на подготовку и раздачу учебных материалов.

Увеличение вовлеченности учеников в учебный процесс через использование современных технологий.

Возможность быстро адаптироваться к изменениям в учебном процессе.

Пример 4: Внедрение Canvas в международном университете

Международный университет внедрил систему Canvas для улучшения управления учебным процессом и интеграции с различными сторонними сервисами. Платформа обеспечила удобный интерфейс для преподавателей и студентов, а также поддержку множества инструментов для улучшения образовательного процесса[36]​​.

**Результаты внедрения**:

Повышение удовлетворенности преподавателей и студентов за счет удобного и интуитивно понятного интерфейса.

Улучшение качества образовательного процесса за счет использования различных инструментов и сервисов.

Возможность легко интегрировать сторонние приложения и сервисы для расширения функциональности системы.

## **1.6. Выводы по Части 1**

На рынке представлено множество систем автоматизации учебного процесса, каждая из которых имеет свои преимущества и недостатки. Примеры успешного внедрения показывают, что правильный выбор системы может значительно улучшить управление учебным процессом, повысить прозрачность и доступность данных, а также сократить время на выполнение рутинных задач. Выбор подходящей системы зависит от специфических нужд и возможностей каждого учебного заведения.

# Часть 2. Проектирование приложения

## **2.1. Техническое задание**

**Цель системы:** Целью разработки информационной системы является автоматизация административных и образовательных процессов в учебном заведении. Это позволит повысить эффективность управления учебным процессом, снизить административную нагрузку и улучшить прозрачность данных о посещаемости и успеваемости студентов.

**Задачи системы:**

**Управление пользователями:**

Обеспечение возможности добавления, редактирования и удаления информации о пользователях системы.

Управление профилями пользователей для упрощения администрирования.

Обеспечение безопасности данных пользователей с использованием хэширования паролей (bcrypt).

**Управление учебными группами:**

Создание и редактирование учебных групп.

Распределение студентов по группам для организации учебного процесса.

**Управление расписанием занятий:**

Создание и редактирование расписания занятий.

Назначение преподавателей и аудиторий для проведения занятий.

**Проведение занятий:**

Обеспечение возможности проведения занятий преподавателями согласно расписанию.

Отметка статуса посещаемости студентов (был, не был, отсутствовал по уважительной причине).

**Создание ведомостей:**

Создание ведомостей для контрольных и экзаменационных работ.

Внесение и хранение данных об оценках студентов.

**Анализ данных о посещаемости и успеваемости:**

Просмотр данных о посещаемости и успеваемости в виде графиков и таблиц.

Обеспечение доступа к аналитическим данным для администраторов и преподавателей системы.

**Защита от несанкционированного доступа:**

Реализация механизма входа в систему с использованием логина и пароля.

Хранение паролей в захэшированном виде для обеспечения безопасности данных.

**Требования к функционалу приложения:**

**Простота использования**: Интерфейс должен быть интуитивно понятным для всех категорий пользователей.

**Надежность**: Приложение должно работать без сбоев и обеспечивать сохранность данных.

**Масштабируемость**: Возможность расширения функционала и добавления новых пользователей без значительных изменений в системе.

**Производительность**: Приложение должно обеспечивать быструю обработку данных и минимальное время отклика.

**Требования к надежности и безопасности:**

**Хранение данных**: Данные должны храниться в защищенной базе данных с регулярным резервным копированием.

**Аутентификация и авторизация**: Вход в систему должен быть защищен с использованием хэширования паролей (bcrypt) и механизмов аутентификации и авторизации.

**Защита данных**: Обеспечение защиты данных от несанкционированного доступа, в том числе с использованием шифрования и регулярных проверок безопасности.

**Выбор технологий:**

**C# и .NET WinForms**: для разработки настольного приложения, предоставляющего весь функционал для администраторов.

**Laravel**: для создания веб-приложения, обеспечивающего доступ преподавателей к системе через браузер.

**MySQL**: для управления базой данных, которая будет хранить всю необходимую информацию о пользователях, студентах, расписаниях, посещаемости и успеваемости.

## **2.2. Структура базы данных**

### 2.2.1. Описание используемых таблиц и полей

Система использует реляционную базу данных для хранения и управления данными, связанными с учебным процессом. База данных реализована с использованием СУБД MySQL. Основные таблицы базы данных и их структура представлены ниже:

Таблица 2.1 сущность control

| **Поле** | **Тип** | **Описание** |
| --- | --- | --- |
| **id** | **int** | **Уникальный идентификатор контроля** |
| **date** | **date** | **Дата проведения контрольного мероприятия** |
| **control\_name** | **varchar(255)** | **Название контрольного мероприятия** |
| **control\_type** | **varchar(255)** | **Тип контрольного мероприятия (система оценивания: "2-5" или "0-100")** |
| **change\_date** | **timestamp** | **Дата последнего изменения записи** |

Таблица 2.2 сущность daily\_schedule

| **Поле** | **Тип** | **Описание** |
| --- | --- | --- |
| **id** | **int** | **Уникальный идентификатор расписания** |
| **schedule\_id** | **int** | **Идентификатор основного расписания** |
| **day** | **tinyint** | **День недели (1 - понедельник, 2 - вторник и т.д.)** |
| **start\_time** | **time** | **Время начала занятия** |
| **end\_time** | **time** | **Время окончания занятия** |
| **change\_date** | **timestamp** | **Дата последнего изменения записи** |

Таблица 2.3 сущность exircise

| **Поле** | **Тип** | **Описание** |
| --- | --- | --- |
| **id** | **int** | **Уникальный идентификатор занятия** |
| **student\_id** | **int** | **Идентификатор студента** |
| **ex\_date** | **date** | **Дата занятия** |
| **status** | **tinyint** | **Статус посещаемости (1 - был, 2 - не был, 3 - отсутствовал по уважительной причине)** |
| **change\_date** | **timestamp** | **Дата последнего изменения записи** |

Таблица 2.4 сущность grades

| **Поле** | **Тип** | **Описание** |
| --- | --- | --- |
| **id** | **int** | **Уникальный идентификатор оценки** |
| **student\_id** | **int** | **Идентификатор студента** |
| **control\_id** | **int** | **Идентификатор контрольного мероприятия** |
| **control\_type** | **varchar(255)** | **Тип контрольного мероприятия (система оценивания: "2-5" или "0-100")** |
| **grade** | **varchar(255)** | **Оценка** |
| **change\_date** | **timestamp** | **Дата последнего изменения записи** |

Таблица 2.5 сущность groups

| **Поле** | **Тип** | **Описание** |
| --- | --- | --- |
| **id** | **int** | **Уникальный идентификатор группы** |
| **group\_name** | **varchar(255)** | **Название группы** |
| **change\_date** | **timestamp** | **Дата последнего изменения записи** |

Таблица 2.6 сущность LoginInfo

| **Поле** | **Тип** | **Описание** |
| --- | --- | --- |
| **id** | **int** | **Уникальный идентификатор записи** |
| **login** | **varchar(255)** | **Логин пользователя** |
| **password** | **varchar(255)** | **Зашифрованный пароль пользователя** |
| **role** | **int** | **Роль пользователя (1 - администратор, 2 - преподаватель)** |
| **user\_data\_id** | **int** | **Идентификатор данных пользователя** |
| **change\_date** | **timestamp** | **Дата последнего изменения записи** |

Таблица 2.7 сущность schedule

| **Поле** | **Тип** | **Описание** |
| --- | --- | --- |
| **id** | **int** | **Уникальный идентификатор расписания** |
| **group\_id** | **int** | **Идентификатор группы** |
| **teacher\_id** | **int** | **Идентификатор преподавателя** |
| **start\_date** | **date** | **Дата начала действия расписания** |
| **end\_date** | **date** | **Дата окончания действия расписания** |
| **change\_date** | **timestamp** | **Дата последнего изменения записи** |

Таблица 2.8 сущность Students

| **Поле** | **Тип** | **Описание** |
| --- | --- | --- |
| **id** | **int** | **Уникальный идентификатор студента** |
| **FIO** | **varchar(255)** | **Полное имя студента** |
| **group\_id** | **int** | **Идентификатор группы** |
| **age** | **int** | **Возраст студента** |
| **spec** | **varchar(255)** | **Специальность студента** |
| **phone** | **varchar(255)** | **Телефон студента** |
| **parent\_info** | **varchar(255)** | **Информация о родителях студента** |
| **start\_date** | **date** | **Дата начала обучения студента** |
| **exercise\_amount** | **int** | **Количество занятий, посещенных студентом** |
| **status** | **tinyint** | **Статус студента (1 - активен, 0 - неактивен)** |
| **change\_date** | **timestamp** | **Дата последнего изменения записи** |

Таблица 2.9 сущность user\_data

| **Поле** | **Тип** | **Описание** |
| --- | --- | --- |
| **id** | **int** | **Уникальный идентификатор записи** |
| **FIO** | **varchar(255)** | **Полное имя пользователя** |
| **phone** | **varchar(255)** | **Телефон пользователя** |
| **post** | **varchar(255)** | **Должность пользователя** |
| **change\_date** | **timestamp** | **Дата последнего изменения записи** |

### 2.2.2. Взаимосвязи между таблицами

В системе присутствуют следующие взаимосвязи между таблицами:

**Таблица** Students связана с таблицей groups через поле group\_id, что позволяет каждому студенту принадлежать к определенной учебной группе.

**Таблица** schedule связана с таблицей groups через поле group\_id и с таблицей user\_data через поле teacher\_id, что позволяет планировать занятия для определенных групп и назначать преподавателей.

**Таблица** exircise связана с таблицей Students через поле student\_id, что позволяет отслеживать посещаемость студентов на занятиях.

**Таблица** grades связана с таблицей Students через поле student\_id и с таблицей control через поле control\_id, что позволяет хранить оценки студентов по различным предметам.

**Таблица** LoginInfo связана с таблицей user\_data через поле user\_data\_id, что позволяет управлять аутентификацией и авторизацией пользователей системы.

Использование связки MySQL и Laravel позволяет легко масштабировать и корректировать базу данных в соответствии с индивидуальными требованиями заказчика. Миграции Laravel обеспечивают удобное управление схемой базы данных, позволяют легко вносить изменения и отслеживать их историю. Это особенно полезно в процессе разработки, когда требования могут меняться, и система должна адаптироваться к этим изменениям.

### 2.2.3. Инфологическая модель предметной области

Инфологическая модель представляет концептуальное описание информационной структуры предметной области, включающей сущности и связи между ними.

Сущности:

**Студенты (Students)**: Представляют информацию о студентах, включая их личные данные и принадлежность к учебным группам.

**Преподаватели (Teachers)**: Представляют информацию о преподавателях, включая их личные данные и специализацию.

**Группы (Groups)**: Представляют учебные группы, к которым принадлежат студенты.

**Расписание (Schedule)**: Представляют расписание занятий, связывая группы и преподавателей.

**Ежедневное расписание (Daily Schedule)**: Детализированное расписание занятий, указывающее дни и время проведения занятий.

**Посещаемость (Exircise)**: Представляют данные о посещаемости студентов на занятиях.

**Оценки (Grades)**: Представляют оценки, полученные студентами за контрольные и экзаменационные работы.

**Контрольные мероприятия (Control)**: Представляют данные о контрольных мероприятиях, таких как экзамены и контрольные работы.

**Информация о пользователях (User Data)**: Содержат информацию о пользователях системы, таких как преподаватели и администраторы.

**Информация для входа (Login Info)**: Содержат данные для аутентификации пользователей, включая логины и захэшированные пароли.

Атрибуты сущностей:

**Студенты (Students)**: id, ФИО, возраст, группа, телефон, информация о родителях, дата начала обучения, количество посещенных занятий, статус.

**Преподаватели (Teachers)**: id, ФИО, телефон, должность.

**Группы (Groups)**: id, название группы.

**Расписание (Schedule)**: id, группа, преподаватель, дата начала, дата окончания.

**Ежедневное расписание (Daily Schedule)**: id, расписание, день недели, время начала, время окончания.

**Посещаемость (Exircise)**: id, студент, дата занятия, статус.

**Оценки (Grades)**: id, студент, контрольное мероприятие, тип оценки, оценка.

**Контрольные мероприятия (Control)**: id, дата, название мероприятия, тип мероприятия.

**Информация о пользователях (User Data)**: id, ФИО, телефон, должность.

**Информация для входа (Login Info)**: id, логин, пароль, роль, id пользователя.

Связи:

**Студенты** принадлежат к **группам**.

**Преподаватели** ведут занятия, связанные с **группами**.

**Оценки** выставляются **студентам** за **контрольные мероприятия**.

**Расписание** связывает **группы** и **преподавателей**.

**Ежедневное расписание** детализирует **расписание** по дням и времени.

**Посещаемость** отражает участие **студентов** в занятиях.

**Информация для входа** связывается с **информацией о пользователях**.

### 2.2.4. Графическое представление базы данных

На следующей диаграмме представлена структура базы данных и взаимосвязи между таблицами:

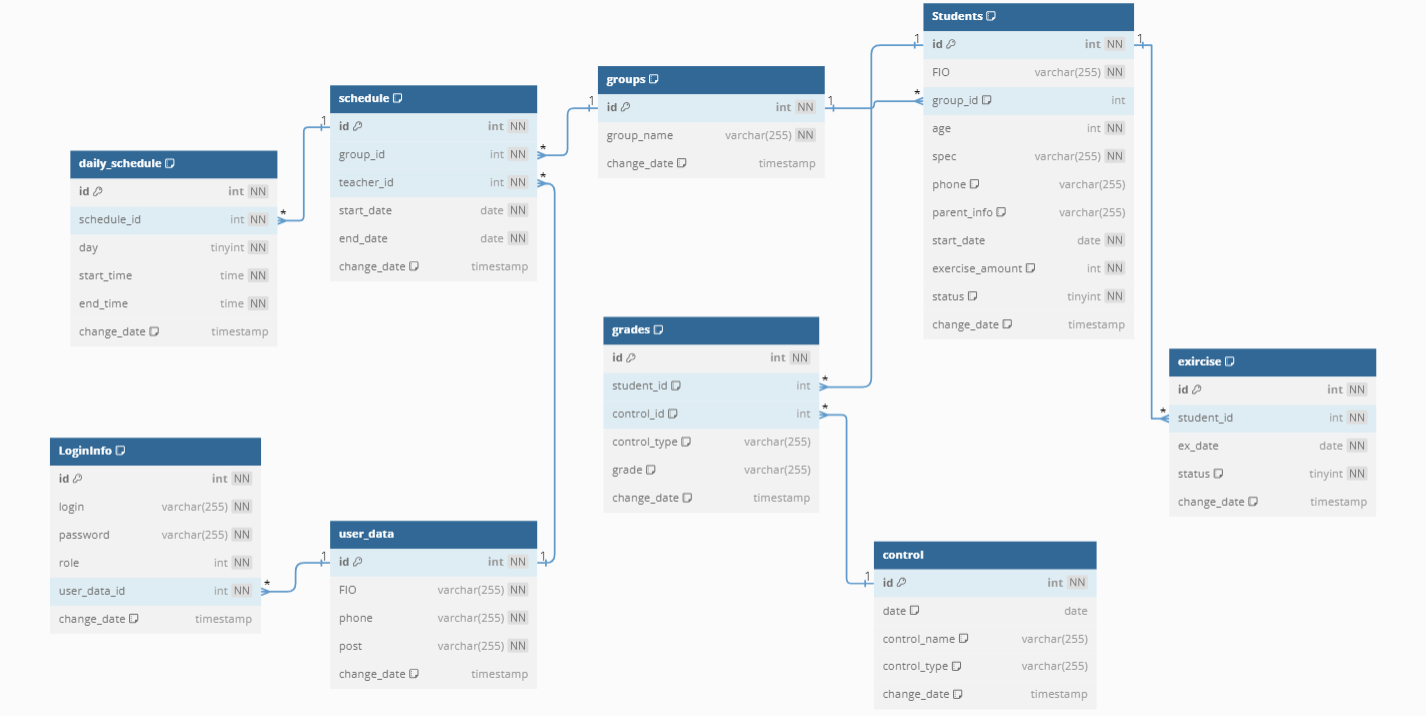


Рисунок 2.1 ER-диаграмма базы данных проектируемой информационной системы

## **2.4. Создание администраторского функционала в настольном приложении на WinForms**

### 2.4.1. Выбор и настройка клиентского окружения

Для создания настольного приложения был выбран фреймворк .NET WinForms, который предоставляет удобные средства для разработки настольных приложений на C#.

**Причины выбора .NET WinForms:**

**Интуитивно понятный интерфейс**: WinForms предоставляет множество инструментов для создания пользовательского интерфейса.

**Гибкость и расширяемость**: Приложение можно легко адаптировать и расширять в соответствии с требованиями.

**Повышенная безопасность**: Размещение административных функций только в настольном приложении снижает риск несанкционированного доступа через интернет.

### 2.4.2. Основные компоненты настольного приложения

**Формы**: Формы в WinForms представляют пользовательский интерфейс и содержат элементы управления, такие как кнопки, текстовые поля и таблицы.

**Обработчики событий**: Обработчики событий управляют взаимодействием пользователя с элементами интерфейса, выполняя соответствующие действия, такие как добавление, редактирование и удаление данных.

### 2.4.3. Управление пользователями и студентами

**Описание функционала:**

* Добавление нового пользователя (преподавателя).
* Редактирование данных существующего пользователя.
* Удаление пользователя.
* Добавление нового студента.
* Редактирование данных существующего студента.
* Удаление студента.

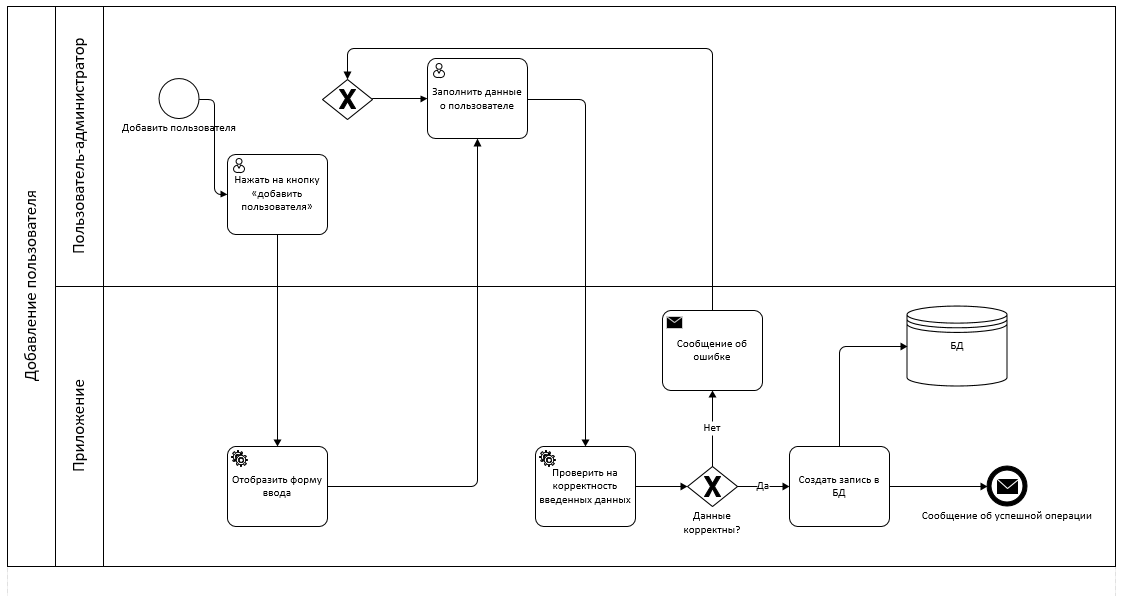


Рисунок 2.2 BPMN-диаграмма процесса добавления пользователя

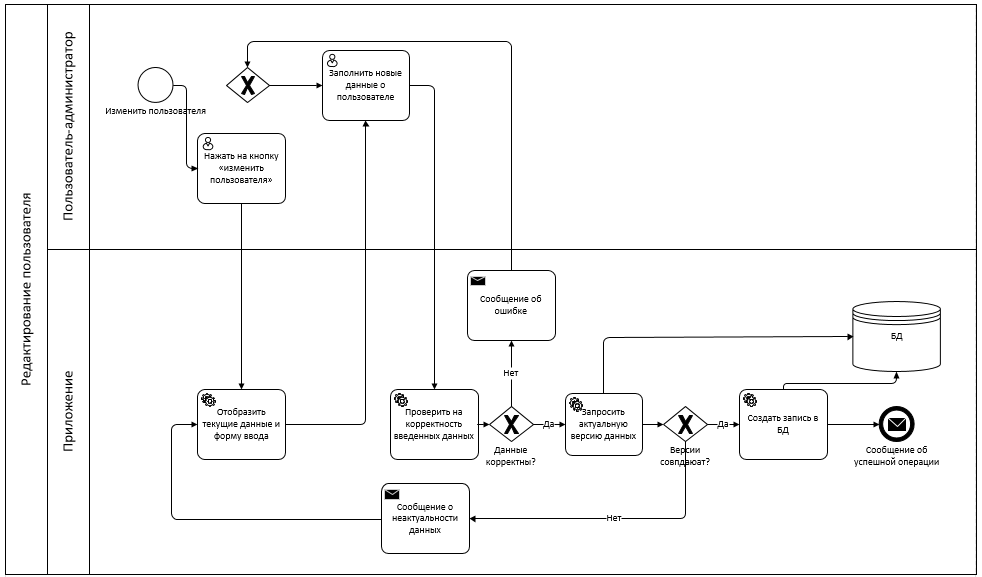


Рисунок 2.3 BPMN-диаграмма процесса редактирования пользователя

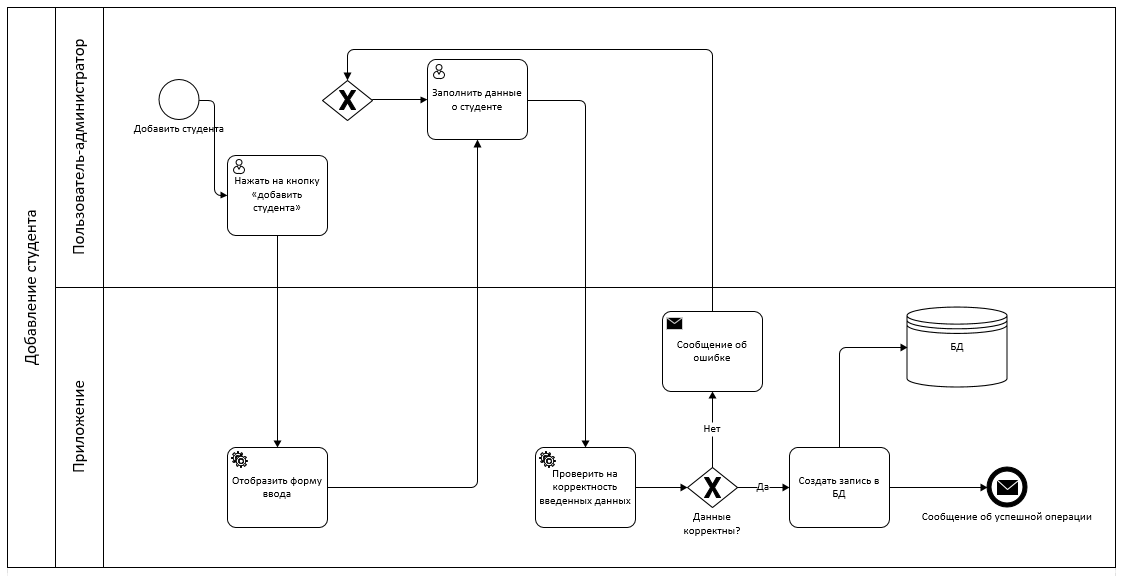


Рисунок 2.4 BPMN-диаграмма процесса создания студента

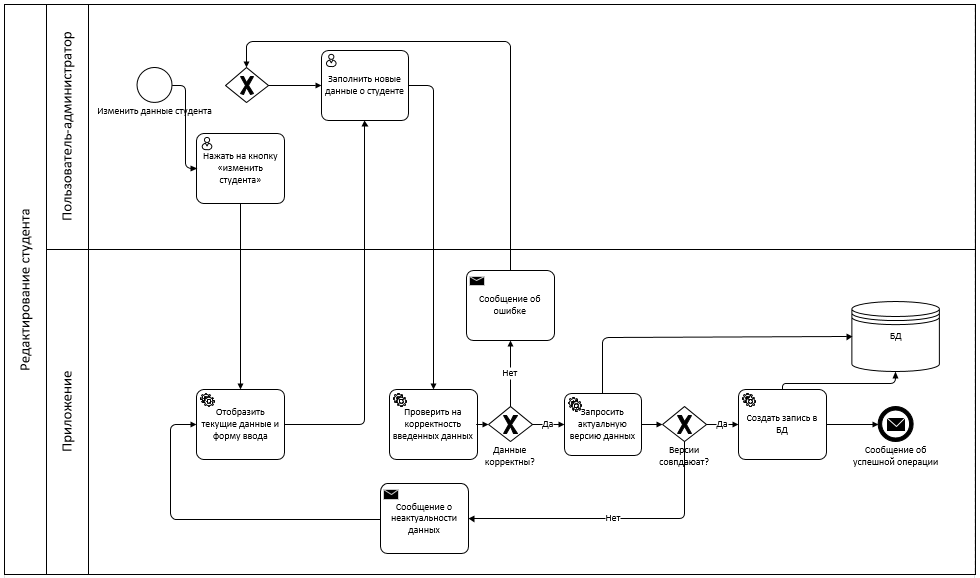


Рисунок 2.5 BPMN-диаграмма процесса редактирования студента

### 2.4.4. Управление учебными группами

**Описание функционала:**

* Добавление новой учебной группы.
* Редактирование состава существующей группы.
* Удаление группы.

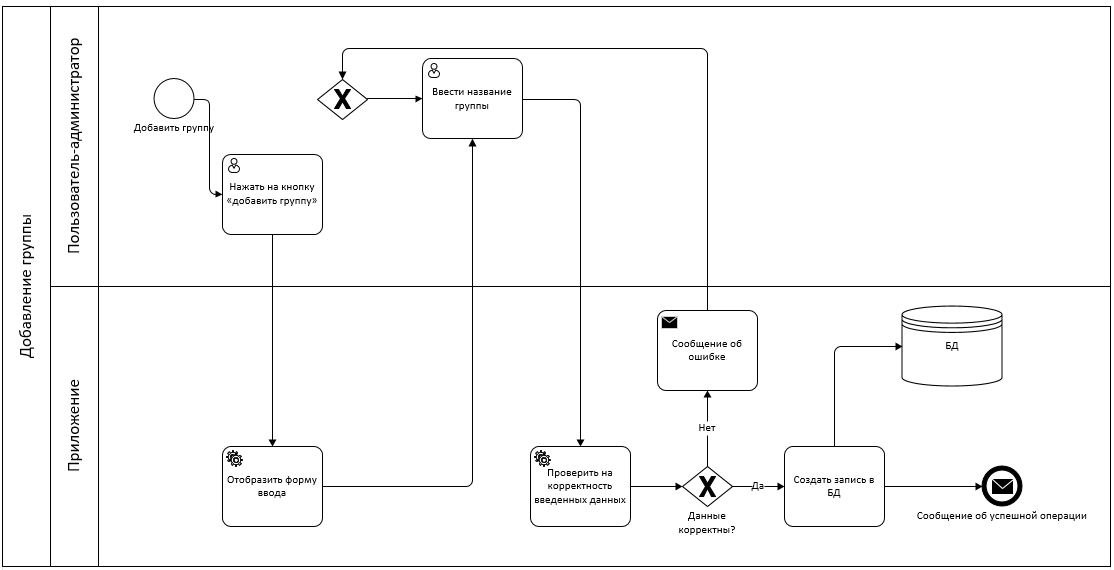


Рисунок 2.6 BPMN-диаграмма процесса добавления группы

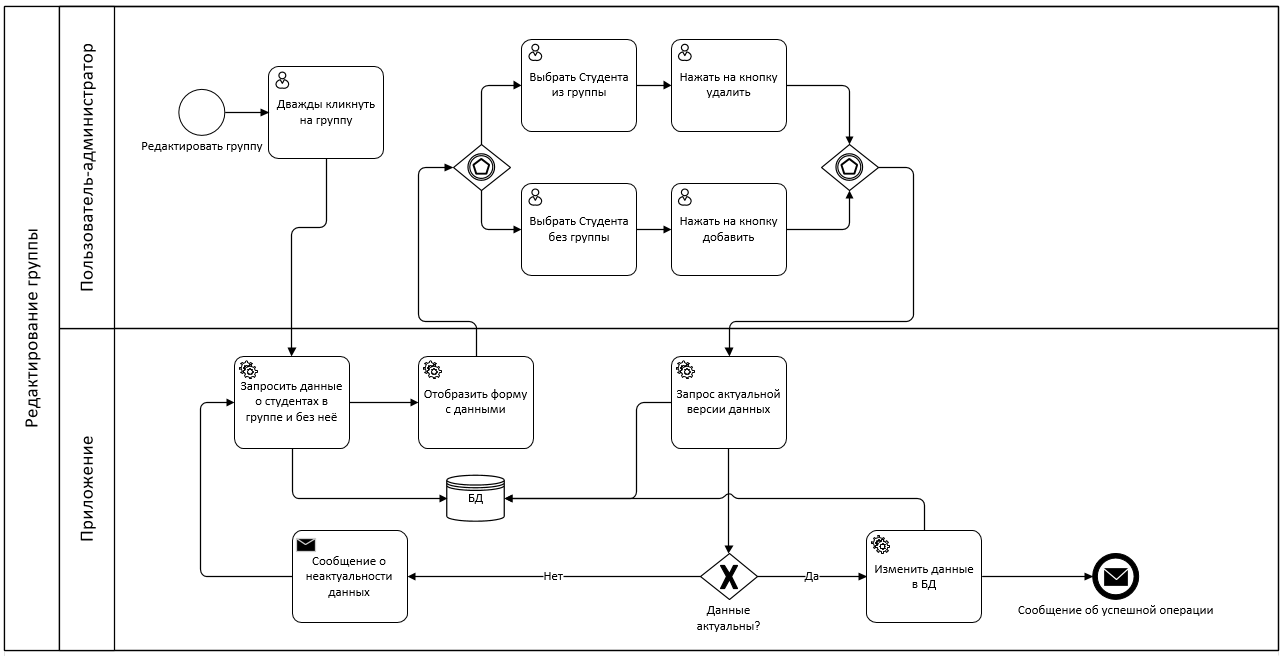


Рисунок 2.7 BPMN-диаграмма процесса редактирования группы

### 2.4.5. Управление расписанием занятий

**Описание функционала:**

* Создание нового расписания занятий.
* Редактирование существующего расписания.

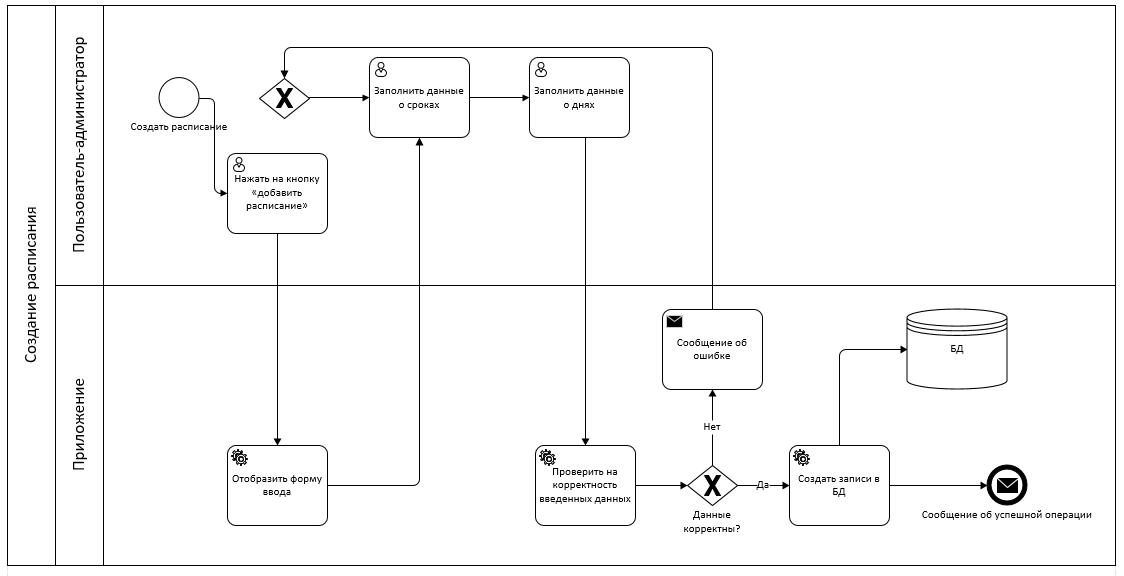


Рисунок 2.8 BPMN-диаграмма процесса создания расписания

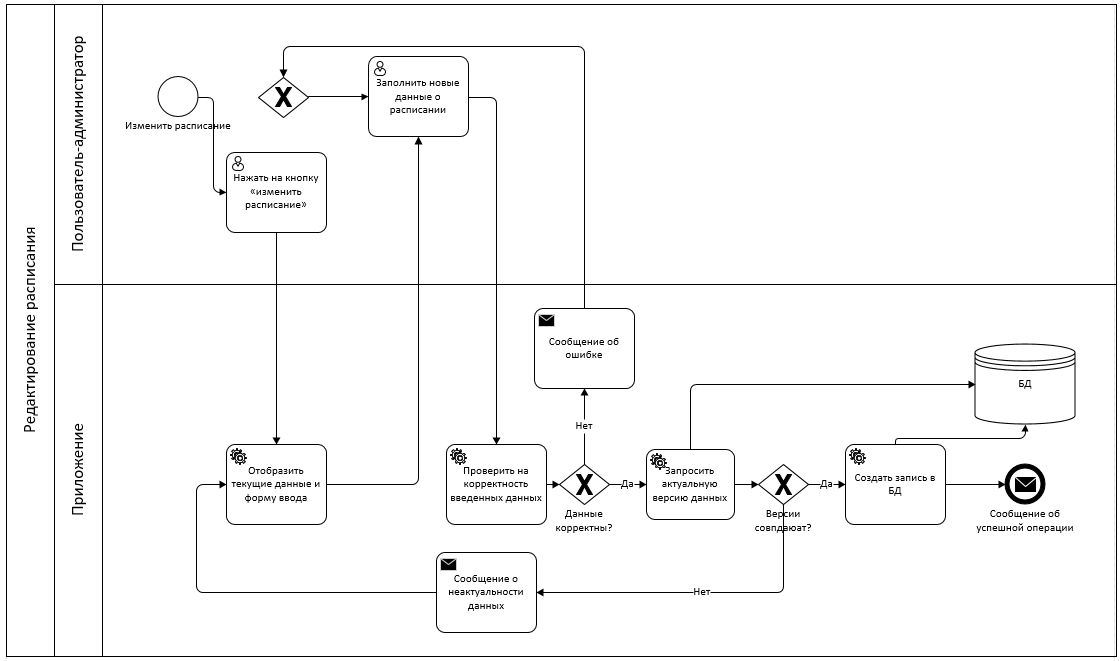


Рисунок 2.9 BPMN-диаграмма процесса редактирования расписания

### 2.4.6. О разделении функционала

Реализация административных функций исключительно в настольном приложении на WinForms обеспечивает повышенную безопасность по сравнению с веб-реализацией.

Настольное приложение работает в локальной сети учреждения, что значительно снижает риск несанкционированного доступа через интернет. Кроме того, хранение данных аутентификации и управление доступом внутри локальной сети обеспечивают дополнительный уровень защиты.

## **2.5. Создание функционала для преподавателей**

### 2.5.1. Функционал в настольном приложении на WinForms

Функционал для преподавателей в настольном приложении на WinForms включает проведение занятий и контрольных работ, просмотр графиков и таблиц с аналитической информацией о посещаемости и успеваемости студентов и групп.

**Описание функционала:**

Проведение занятий и отметка посещаемости студентов.

Создание ведомостей для контрольных и экзаменационных работ.

Просмотр и анализ данных о посещаемости и успеваемости студентов.

### 2.5.2. Функционал в веб-приложении на Laravel

Для серверной части приложения был выбран фреймворк Laravel, который представляет собой мощный инструмент для создания веб-приложений с удобной архитектурой и множеством встроенных функций.

**Причины выбора Laravel:**

**Мощная архитектура**: Laravel основан на MVC (Model-View-Controller) паттерне, что позволяет структурировать код и разделять логику приложения.

**Инструменты для работы с базой данных**: В Laravel встроена система миграций, которая упрощает создание и изменение структуры базы данных.

**Безопасность**: Laravel предоставляет встроенные средства для обеспечения безопасности, такие как хэширование паролей и защита от CSRF атак.

**Сообщество и документация**: Laravel имеет обширное сообщество и хорошую документацию, что облегчает разработку и поддержку приложения.

**Основные компоненты серверной части:**

**Модели**: Модели в Laravel представляют данные и логику работы с ними. Каждая модель соответствует таблице в базе данных и определяет взаимодействие с этой таблицей.

**Контроллеры**: Контроллеры содержат логику обработки запросов, поступающих от клиента, и взаимодействие с моделями. Они управляют потоком данных между моделью и представлением.

**Маршруты**: Маршруты определяют, какие действия должны выполняться при обращении к определенным URL. В Laravel маршруты определяются в файлах routes/web.php и routes/api.php.

**Описание функционала:**

* Проведение занятий и отметка посещаемости студентов.
* Создание ведомостей для контрольных и экзаменационных работ.
* Просмотр расписания занятий.
* Просмотр и анализ данных о посещаемости и успеваемости студентов.

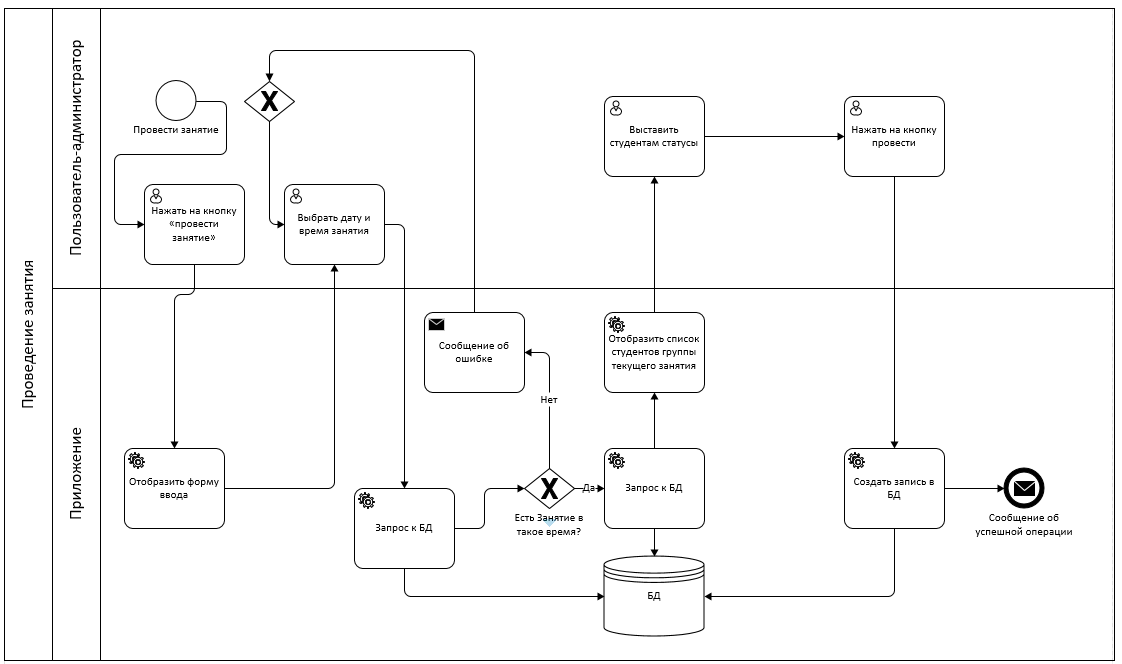


Рисунок 2.10 BPMN-диаграмма процесса проведения занятия

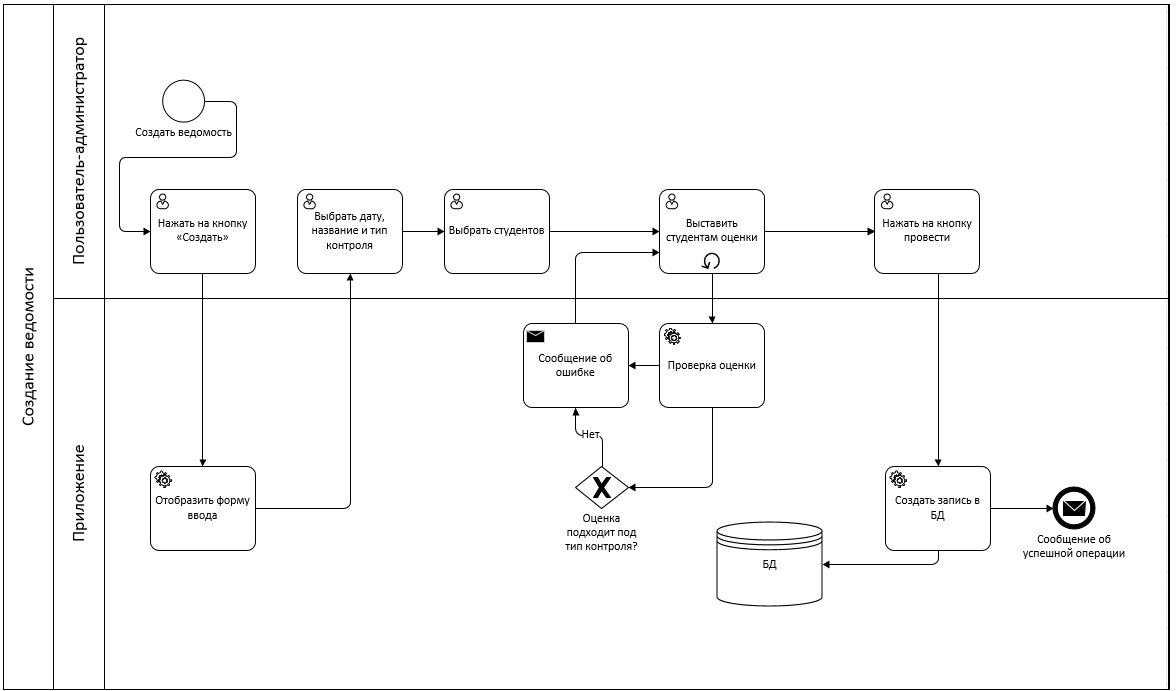


Рисунок 2.11 BPMN-диаграмма процесса создания ведомости

## **2.6. Анализ данных о посещаемости и успеваемости**

Анализ данных о посещаемости и успеваемости студентов является важной частью системы, так как позволяет администраторам и преподавателям получать актуальную информацию о прогрессе студентов и принимать обоснованные решения для улучшения учебного процесса.

### 2.6.1. Цели анализа данных

Предоставление информации о посещаемости студентов для выявления проблемных областей.

Мониторинг академической успеваемости студентов для своевременного вмешательства в случае необходимости.

Создание комплексных графиков для визуального представления данных, что облегчает понимание и интерпретацию информации.

### 2.6.2. Методы анализа данных

Для анализа данных используются различные методы и инструменты, включая SQL-запросы для извлечения данных из базы данных, а также инструменты для визуализации данных, такие как графики и диаграммы.

**Примеры SQL-запросов:**

**Запрос для получения данных о посещаемости студентов:**

SELECT s.FIO, e.ex\_date, e.status FROM Students s JOIN exircise e ON s.id = e.student\_id WHERE e.ex\_date BETWEEN '2023-01-01' AND '2023-12-31' ORDER BY s.FIO, e.ex\_date;

**Запрос для получения данных об успеваемости студентов:**

SELECT s.FIO, c.control\_name, g.grade FROM Students s JOIN grades g ON s.id = g.student\_id JOIN control c ON g.control\_id = c.id WHERE c.control\_type = 'exam' ORDER BY s.FIO, c.control\_name;

### 2.6.3. Визуализация данных

Визуализация данных является ключевым элементом анализа, так как позволяет пользователям легко интерпретировать информацию. В системе реализована возможность создания комплексных графиков для отображения данных о посещаемости и успеваемости.

**Инструменты для визуализации:**

**Для настольного приложения на WinForms:** используются встроенные компоненты .NET для создания графиков и диаграмм.

**Для веб-приложения на Laravel:** используется JavaScript-библиотека Chart.js для создания интерактивных графиков.

**Примеры визуализации:**

**Визуализация данных о посещаемости:**

**График посещаемости групп:**

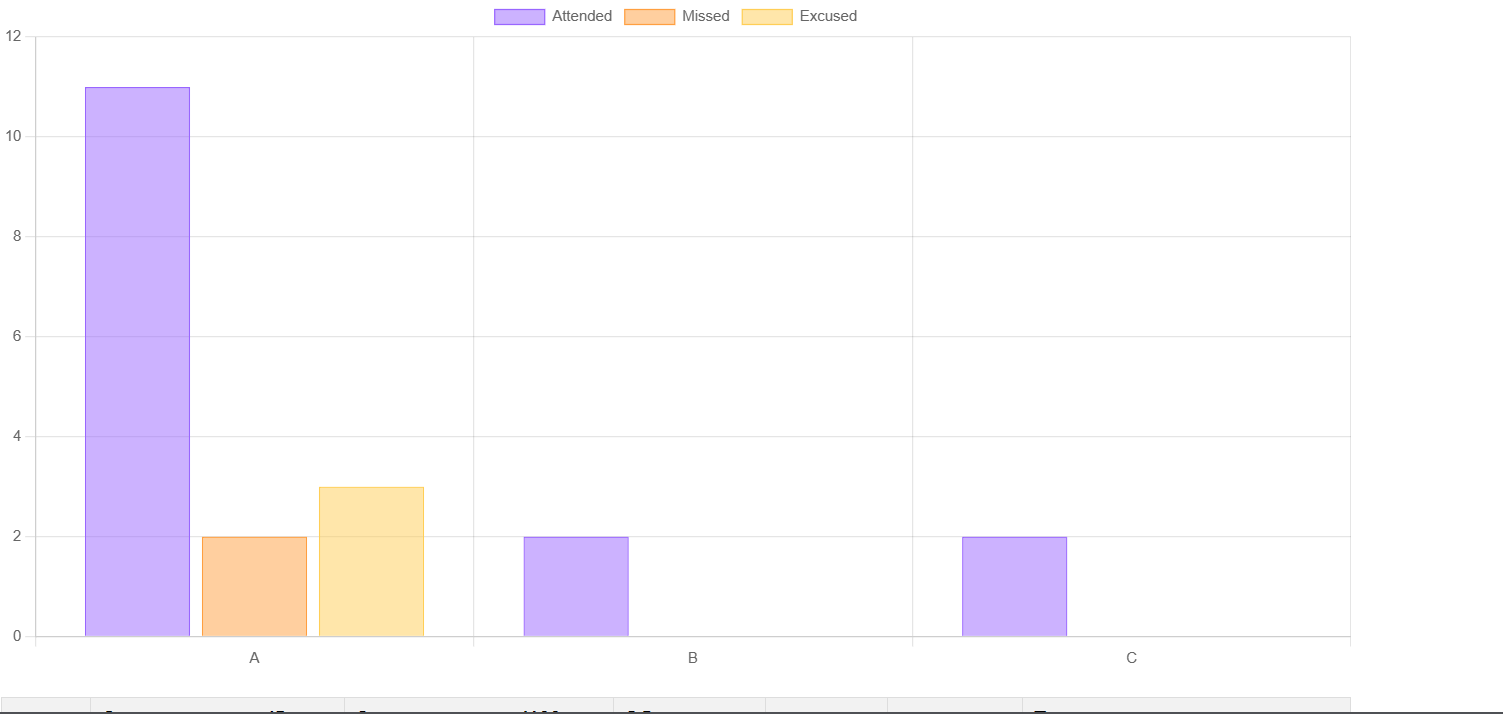


Рисунок 2.12 График посещаемости групп

Ось X: посещения

Ось Y: группы

**Визуализация данных об успеваемости:**

**График распределения оценок внутри группы:**

Ось X: средняя оценка по пятибальной шкале

Ось Y: средняя оценка по стобальной шкале

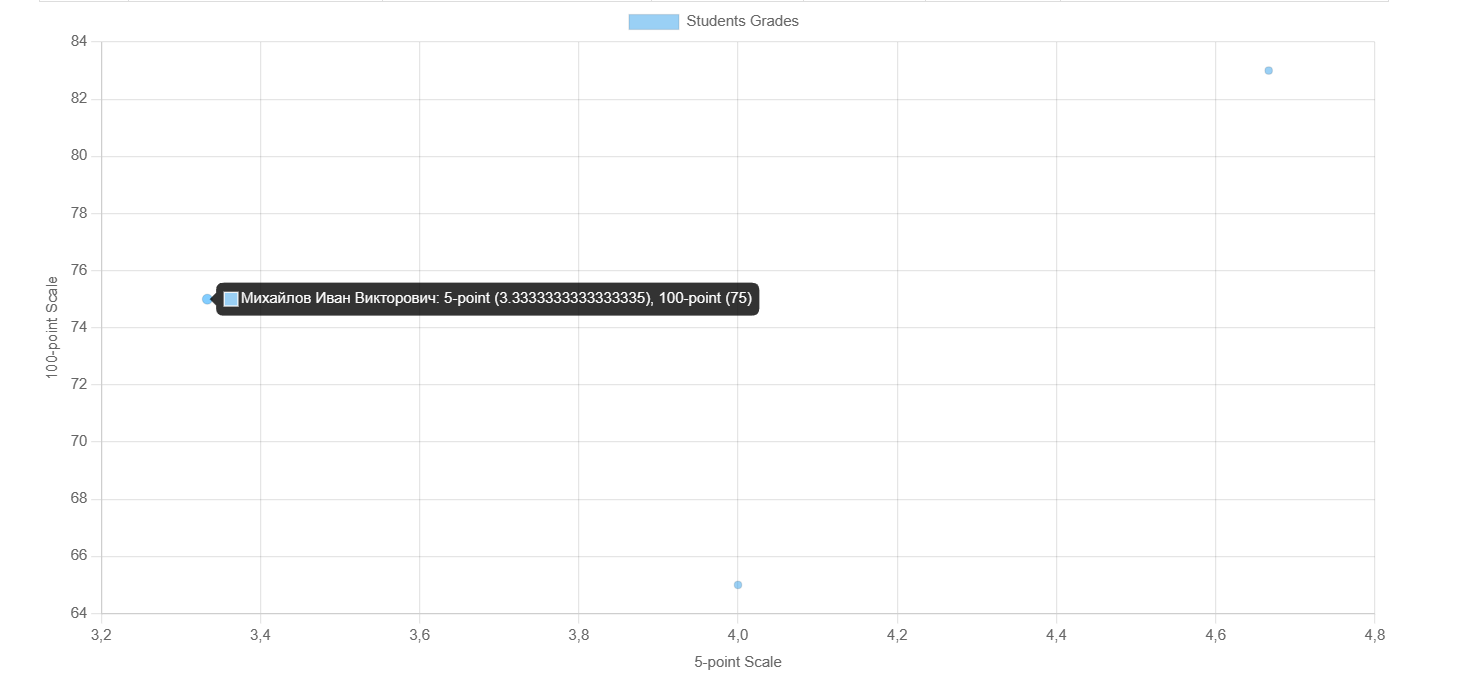


Рисунок 2.13 График распределения оценок внутри группы

**Комплексные графики:**

**График динамики среднего балла групп:**

Ось X: Даты

Ось Y: Средний балл группы

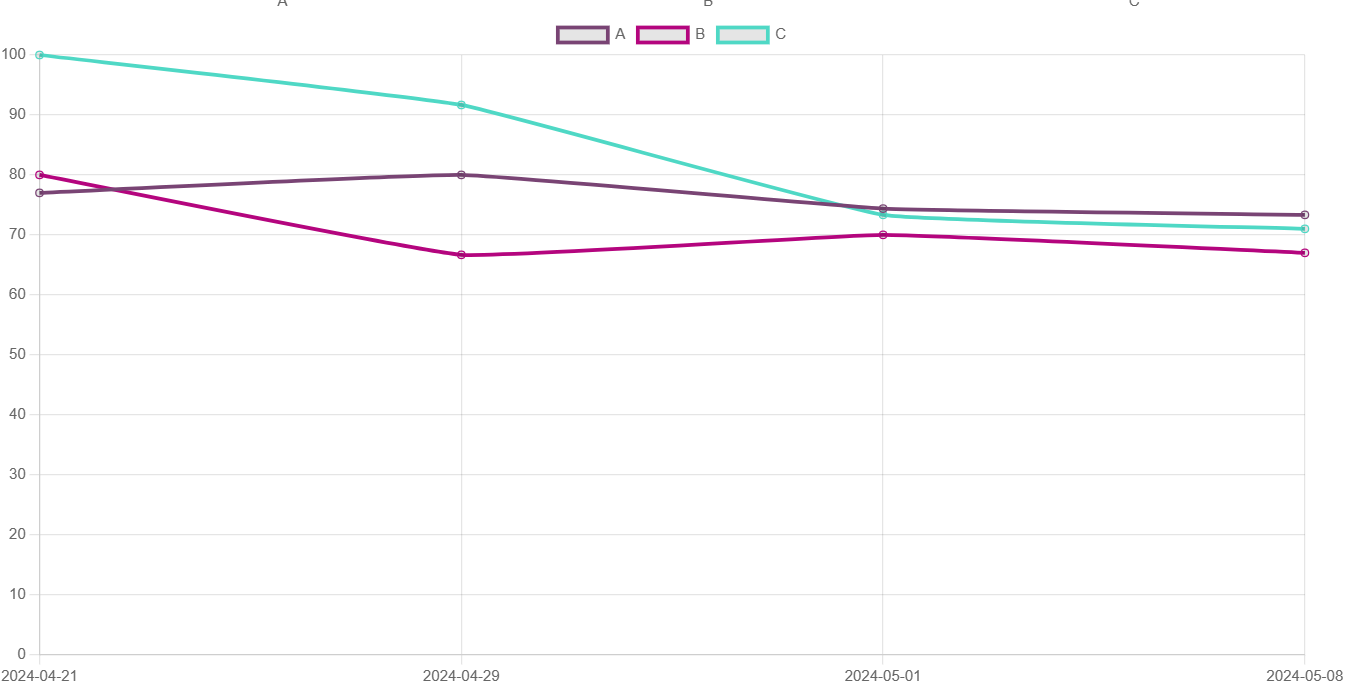


Рисунок 2.14 Динамика среднего балла групп с течением времени

**График успеваемости конкретного студента:**

Ось X: Даты

Ось Y: средняя оценка студента



Рисунок 2.15 График динамики оценки студента

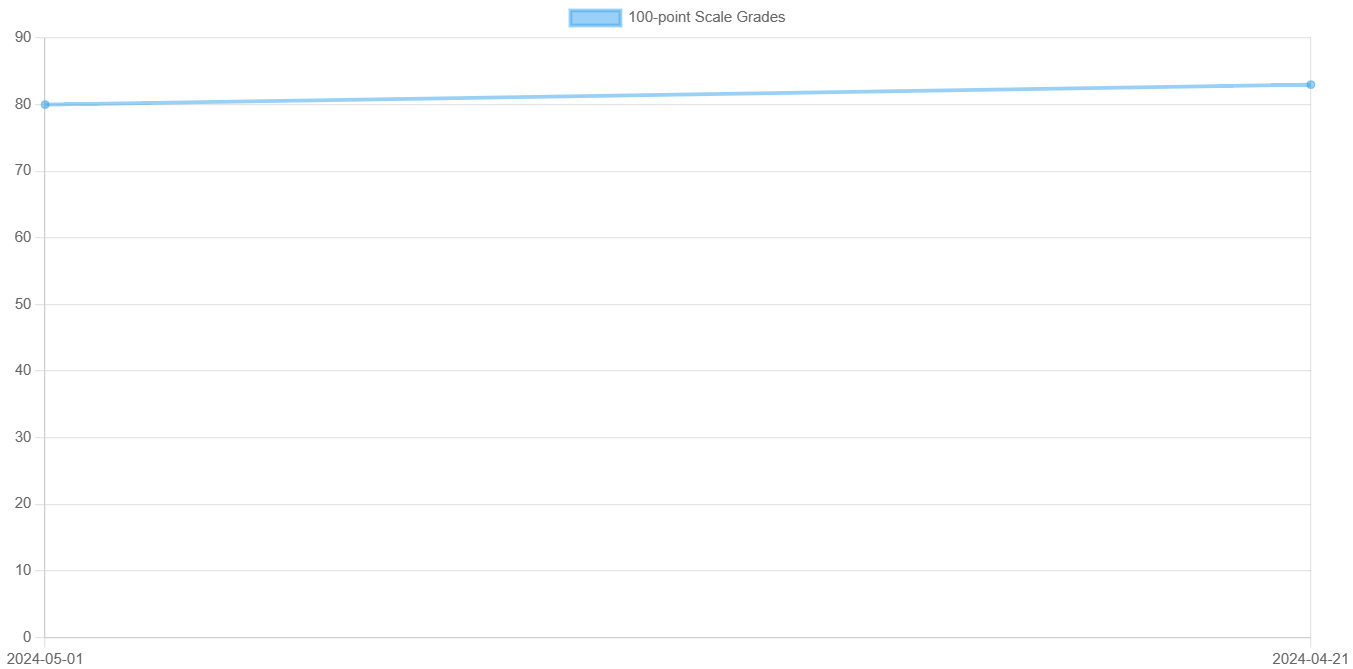


Рисунок 2.16 График динамики оценки студента

### 2.6.4. Значимость комплексных графиков

Комплексные графики играют важную роль в анализе данных, так как они предоставляют наглядное представление о динамике учебного процесса и позволяют легко выявить тенденции и проблемные области.

**Примеры использования комплексных графиков:**

**Отслеживание прогресса группы**: Графики динамики среднего балла группы позволяют преподавателям и администраторам видеть, как изменяется успеваемость группы в течение времени, и принимать меры для улучшения ситуации.

**Индивидуальный подход к студентам**: Графики успеваемости конкретного студента помогают преподавателям выявлять студентов, которые нуждаются в дополнительной поддержке, и разрабатывать индивидуальные планы обучения.

## **2.7. Защита от несанкционированного доступа**

### 2.7.1. Механизмы аутентификации и авторизации

Для обеспечения безопасности доступа к системе используются следующие механизмы аутентификации и авторизации:

**Аутентификация**:

**Логины и пароли**: Каждый пользователь системы имеет уникальный логин и пароль для входа в систему.

**Хэширование паролей**: Пароли пользователей хранятся в захэшированном виде с использованием алгоритма bcrypt, что предотвращает их расшифровку в случае утечки данных.

**Авторизация**:

**Роли пользователей**: В системе предусмотрены две роли — администратор и преподаватель. Администраторы имеют полный доступ ко всем функциям системы, в то время как преподаватели имеют доступ только к функциям, связанным с проведением занятий и контролем успеваемости.

**Контроль доступа**: Доступ к различным функциям системы ограничен в зависимости от роли пользователя.

### 2.7.2. Хранение паролей в захэшированном виде (bcrypt)

Для обеспечения безопасности хранения паролей используется алгоритм bcrypt, который обеспечивает надежное хэширование данных. При этом каждый пароль хэшируется с использованием уникальной соли, что делает невозможным использование предрассчитанных таблиц (rainbow tables) для взлома паролей.

**Пример кода на Laravel**:

// Обработать данные формы входа

    public function login(Request $request)

{

    $credentials = $request->only('login', 'password');

    if (Auth::attempt($credentials)) {

        $request->session()->regenerate();

        return redirect()->intended('home'); // Перенаправление на главную страницу

    }

    return back()->withErrors([

        'login' => 'The provided credentials do not match our records.',

    ]);

**Пример кода на C# (WinForms)**:

string salt = BCrypt.Net.BCrypt.GenerateSalt(10, 'y');

cmd.CommandText = "INSERT INTO LoginInfo (login, password, role, user\_data\_id) VALUES (@login, @password, @role, @user\_data\_id)";

cmd.Parameters.Clear();

cmd.Parameters.AddWithValue("@user\_data\_id", user\_id);

cmd.Parameters.AddWithValue("@login", this.age\_ch.Text.ToString());

cmd.Parameters.AddWithValue("@password", BCrypt.Net.BCrypt.HashPassword(this.spec\_ch.Text.ToString(),salt).ToString());

cmd.Parameters.AddWithValue("@role", this.ex\_amount\_ch.Text.ToString());

conn.Open();

cmd.ExecuteNonQuery();

conn.Close();

cmd.Parameters.Clear();

this.Close();

## **2.8. Выводы по Части 2**

В данной главе представлено техническое задание, описан процесс проектирования и разработки информационной системы для автоматизации администрирования и мониторинга учебного процесса. Был рассмотрен выбор технологий, включая использование C# и .NET WinForms для настольного приложения, Laravel для веб-приложения и MySQL для базы данных.

Детально описана структура базы данных, включая основные таблицы и взаимосвязи между ними. Проанализированы методы обеспечения безопасности данных, такие как хэширование паролей и механизмы аутентификации и авторизации. Также подробно изложен процесс создания и настройки пользовательского интерфейса для различных ролей пользователей (администраторов и преподавателей).

В ходе проектирования и разработки системы были достигнуты следующие результаты:

Разработана структура базы данных, обеспечивающая эффективное хранение и управление данными о студентах, преподавателях, учебных группах и расписаниях.

Реализованы основные функции для управления пользователями, студентами, учебными группами и расписаниями в настольном приложении на WinForms.

Разработано веб-приложение на Laravel, обеспечивающее функции для проведения занятий, контроля успеваемости и анализа данных.

Обеспечена защита данных с использованием современных методов аутентификации, авторизации и хэширования паролей.

В целом, разработанная система позволяет автоматизировать административные и мониторинговые задачи в учебных заведениях, что способствует повышению эффективности управления учебным процессом и улучшению качества образовательных услуг.

# Часть 3. Описание разработанного приложения

## **3.1. Руководство пользователя**

### 3.1.1. Описание интерфейса приложения

**Интерфейс настольного приложения (WinForms)**:

**Главное окно**: Содержит меню для навигации по основным разделам системы.

**Раздел "Студенты"**: Позволяет добавлять, редактировать и удалять информацию о студентах.

**Раздел "Пользователи"**: Позволяет управлять учетными записями преподавателей.

**Раздел "Группы"**: Позволяет создавать и редактировать учебные группы.

**Раздел "Расписание"**: Позволяет создавать и редактировать расписание занятий.

**Раздел "Занятия"**: Позволяет преподавателям проводить занятия и отмечать посещаемость студентов.

**Раздел "Контрольные работы"**: Позволяет создавать и редактировать ведомости для контрольных и экзаменационных работ.

**Раздел "Расписание"**: Отображает расписание занятий.

**Раздел "Анализ данных"**: Предоставляет инструменты для анализа данных о посещаемости и успеваемости.

**Интерфейс веб-приложения (Laravel)**:

**Главная страница**: Отображает аналитическую информацию о группах и студентах

**Раздел "Занятия"**: Позволяет преподавателям проводить занятия и отмечать посещаемость студентов.

**Раздел "Контрольные работы"**: Позволяет создавать и редактировать ведомости для контрольных и экзаменационных работ.

**Раздел "Расписание"**: Отображает расписание занятий.

### 3.1.2. Основные функции и как ими пользоваться

**Добавление студента (WinForms)**:

* Перейдите в раздел "Студенты".
* Нажмите кнопку "Добавить".
* Заполните форму с данными студента и нажмите "Сохранить".

**Редактирование данных студента (WinForms)**:

* Перейдите в раздел "Студенты".
* Выберите студента из списка и нажмите кнопку "Редактировать".
* Внесите необходимые изменения и нажмите "Сохранить".

**Проведение занятия (Laravel/WinForms)**:

* Перейдите в раздел "Занятия".
* Выберите Дату и время занятия.
* Отметьте присутствие студентов и нажмите "Сохранить".

**Создание ведомости (Laravel/WinForms)**:

* Перейдите в раздел "Контрольные работы".
* Нажмите кнопку "Создать ".
* Выберите дату, название и систему оценивания контрольной работы.
* Добавьте необходимых студентов из списка
* Выставьте соответствующие студентам оценки и нажмите "Сохранить".

### 3.1.3. Примеры сценариев использования

Сценарий 1: Добавление нового студента

* Администратор открывает настольное приложение и переходит в раздел "Студенты".
* Нажимает кнопку "Добавить".
* Заполняет форму с данными студента и сохраняет изменения.

Сценарий 2: Проведение занятия и отметка посещаемости

* Преподаватель входит в веб-приложение и переходит в раздел "Занятия".
* Выбирает занятие из списка и нажимает "Провести".
* Отмечает присутствие студентов и сохраняет изменения.

## **3.2. Тестирование приложения**

### 3.2.1 Методики тестирования

Для тестирования приложения использовались следующие методики:

**Функциональное тестирование:** Проверка корректности выполнения всех функций приложения.

**Регрессионное тестирование:** Проверка работы системы после внесения изменений и обновлений.

**Тестирование производительности:** Оценка времени отклика и нагрузки на систему при выполнении основных операций.

**Статический анализ кода:** Использование инструментов для проверки качества и корректности исходного кода.

**Анализ цикломатической сложности:** Оценка сложности программного кода.

### 3.2.2 Предотвращение некорректного использования

Для предотвращения некорректного использования приложения и обеспечения надежности и безопасности данных были предприняты следующие меры:

**Проверка корректности введенных данных:**

Каждая функция, которая вносит, изменяет или удаляет данные в базе, обязательно выполняет проверки на корректность введенных данных. Например, проверяются форматы дат и их адекватность, форматы числовых значений и строковых данных.

Используются регулярные выражения для проверки правильности ввода данных, таких как электронная почта, номера телефонов и прочие специфические форматы.

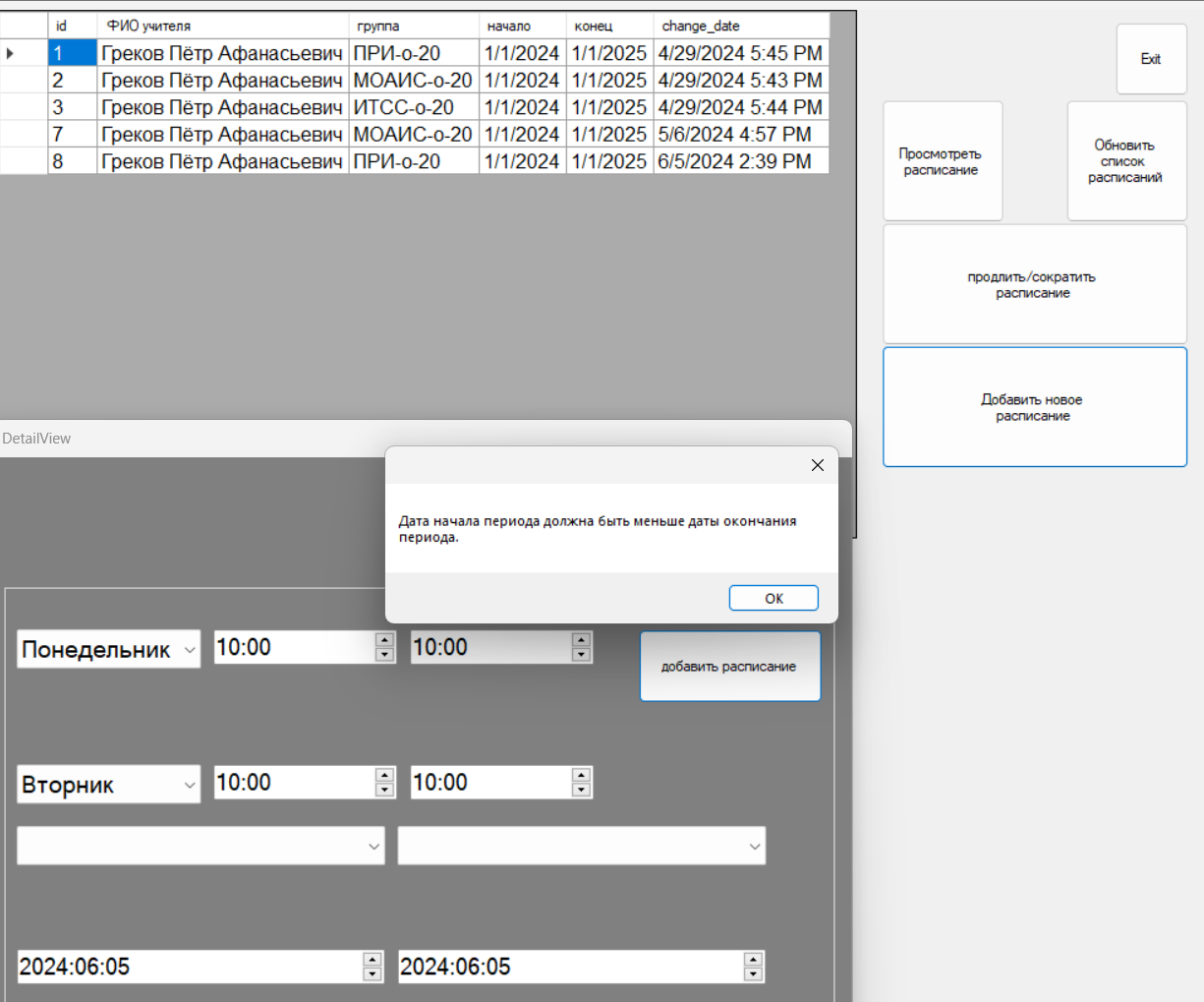


Рисунок 3.1 Пример обработки введения некорректного временного периода длительности расписания

**Проверка актуальности данных:**

Перед изменением или удалением данных выполняется проверка на актуальность предоставленных пользователю данных по сравнению с базой данных. Это предотвращает ситуацию, когда один пользователь пытается изменить данные, которые уже были изменены другим пользователем.

Реализована система версионирования данных, которая позволяет отслеживать изменения и предотвращает конфликты при одновременной работе нескольких пользователей.

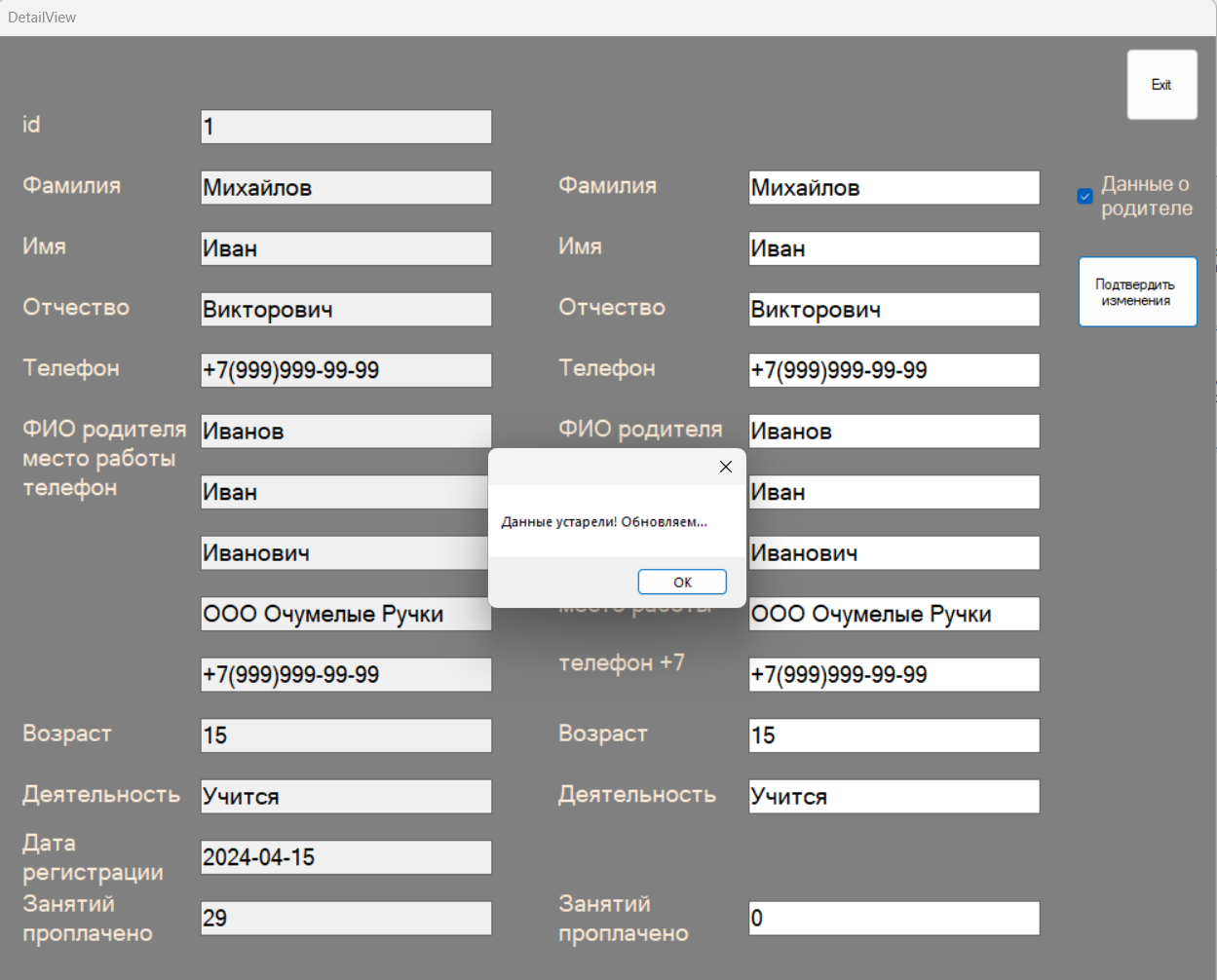


Рисунок 3.2 Пример обработки ошибки об устаревших данных при изменении информации о студенте

**Механизмы транзакций:**

Для обеспечения целостности данных и предотвращения их потери при сбоях используются транзакции. Все операции по изменению данных оборачиваются в транзакции, которые гарантируют либо полное выполнение всех операций, либо их отмену в случае ошибки.

**Управление доступом:**

Реализованы роли и права доступа, которые ограничивают доступ к функциям приложения в зависимости от уровня пользователя (администратор, преподаватель). Это предотвращает некорректное использование функций приложения неавторизованными пользователями.

**Тестирование производительности:**

Проводилось тестирование производительности для оценки времени отклика системы при выполнении основных операций и нагрузок на сервер. Результаты тестирования позволили оптимизировать критические участки кода и базы данных, обеспечив стабильную работу приложения под нагрузкой.

## **3.3 Анализ полученных результатов и оценка работы приложения**

Во время тестирования готового программного продукта были проанализированы и оценены результаты работы приложения. Программа успешно справляется с обработкой большого объема данных, что позволяет сократить время, затрачиваемое на административные задачи, и повышает удобство для пользователей.

Для проведения анализа и оценки качества кода были использованы следующие инструменты:

SonarQube: для статического анализа кода.

Coverlet: для определения уровня покрытия тестами.

NDepend: для оценки цикломатической сложности, метрик поддерживаемости и расчета метрики Холстеда.

Результаты тестов:

Таблица 3.1 — Результаты статического анализа кода с помощью SonarQube

| **Код** | **Описание ошибки** | **Кол-во ошибок** |
| --- | --- | --- |
| 0 | Ошибок нет | 0 |
| 1 | Обнаружено нарушение "Fatal" | 0 |
| 2 | Обнаружено нарушение "Error" | 3 |
| 4 | Обнаружено нарушение "Warning" | 12 |
| 8 | Обнаружено нарушение "Refactor" | 15 |
| 16 | Обнаружено нарушение "Convention" | 20 |

Таблица 3.2 — Определение уровня покрытия тестами с помощью Coverlet

| **Модуль** | **Операторы** | **Пропущено** | **Покрытие** |
| --- | --- | --- | --- |
| DetailView.cs | 523 | 112 | 79% |
| EnterForm.cs | 278 | 34 | 88% |
| ExtraDetailView.cs | 412 | 55 | 87% |
| LogicForm.cs | 367 | 60 | 84% |
| MainForm.cs | 723 | 145 | 80% |
| Program.cs | 189 | 12 | 94% |
| StatisticsForm.cs | 409 | 51 | 87% |

Таблица 3.3 — Расчет метрики Холстеда с помощью NDepend

| **Метрика** | **Значение** |
| --- | --- |
| Словарь уникальных операндов | 245 |
| Общее число операндов | 680 |
| Словарь программы | 350 |
| Длина программы | 1500 |
| Объем программы | 4200 |
| Сложность программы | 8.5 |
| Уровень ошибок | 0.75 |

Приложение продемонстрировало хорошую производительность и надежность, что делает его пригодным для использования в учебных заведениях. Тем не менее, были выявлены недостатки в некоторых частях кода, которые нуждаются в оптимизации и рефакторинге. Эти проблемы могут быть решены в дальнейшем за счет улучшения кода и увеличения покрытия тестами.

## **3.4. Выводы по части 3**

В данной главе представлено руководство пользователя, описан процесс тестирования и анализа результатов работы приложения. Система мониторинга и администрирования учебного процесса показала свою эффективность и удобство в использовании, а также высокий уровень безопасности данных. Разработанная система позволяет существенно повысить эффективность управления учебным проце за счет автоматизации рутинных процессов и предоставления аналитической информации для принятия решений.

# Заключение

Главным результатом ВКР является готовый программный продукт, который осуществляет автоматизацию администрирования и мониторинга образовательного проуесса учебным заведением. В результате, время на выполнение административных и мониторинговых задач сократилось, повысилась эффективность управления и улучшилось качество образовательных услуг.

При разработке использовались языки программирования C# и PHP, фреймворки .NET и Laravel, а также MySQL для создания базы данных. Приложение включает в себя настольную и веб-версии, обеспечивая удобный доступ для различных категорий пользователей.

В процессе работы были достигнуты следующие результаты:

Изучены современные методы автоматизации образовательных процессов.

Рассмотрены и применены различные технологии для создания многофункционального приложения.

Создана и протестирована информационная система для учебных заведений.

Разработаны пользовательские интерфейсы для настольного и веб-приложения.

Спроектирована база данных для эффективного хранения и управления информацией.

Тестирование программного продукта показало высокую производительность и надежность системы.

Приложение продемонстрировало свою пригодность для использования в образовательных учреждениях. Тем не менее, были выявлены некоторые области для улучшения, такие как расширение функциональности и интеграция с другими системами.

# Список использованной литературы

1. Garcia-Molina, H., Ullman, J. D., & Widom, J. Database Systems: The Complete Book / H. Garcia-Molina, J. D. Ullman, J. Widom. — Pearson, 2008.
2. Reyes, M., Ramos, F., & Marti, L. Education and Information Technologies / M. Reyes, F. Ramos, L. Marti. — Springer, 2015.
3. O'Brien, J. A., & Marakas, G. M. Management Information Systems / J. A. O'Brien, G. M. Marakas. — McGraw-Hill, 2011.
4. Connolly, T., & Begg, C. Database Systems: A Practical Approach to Design, Implementation, and Management / T. Connolly, C. Begg. — Pearson, 2014.
5. Laudon, K. C., & Laudon, J. P. Management Information Systems: Managing the Digital Firm / K. C. Laudon, J. P. Laudon. — Pearson, 2019.
6. Stair, R. M., & Reynolds, G. W. Principles of Information Systems / R. M. Stair, G. W. Reynolds. — Cengage Learning, 2017.
7. Kroenke, D. M., & Auer, D. J. Database Concepts / D. M. Kroenke, D. J. Auer. — Pearson, 2013.
8. Shelly, G. B., & Rosenblatt, H. J. Systems Analysis and Design / G. B. Shelly, H. J. Rosenblatt. — Cengage Learning, 2012.
9. Oz, E. Management Information Systems / E. Oz. — Cengage Learning, 2009.
10. Разработка модуля формирования учебных поручений в вузе на платформе "1С:Предприятие" // Проблемы и перспективы прикладных исследований. — 2020. — № 1(12). — С. 45-53. Режим доступа: <https://applied-research.ru/ru/article/view?id=13575>.
11. Система "1С:Предприятие" в развитии вуза // Наука и образование: новое время. — 2021. — № 4(8). — С. 77-84. Режим доступа: <https://science-education.ru/ru/article/view?id=18645>.
12. Разработка информационной модели учебного процесса кафедры вуза на платформе "1С:Предприятие" // Технологии высшего образования. — 2021. — № 2(15). — С. 102-109. Режим доступа: <https://top-technologies.ru/ru/article/view?id=39198>.
13. Направления автоматизации образовательной деятельности // Информационные технологии в образовании. — 2020. — № 5(20). — С. 98-105. Режим доступа: <https://cyberleninka.ru/article/n/napravleniya-avtomatizatsii-obrazovatelnoy-deyatelnosti>.
14. Dougiamas, M., & Taylor, P. C. Moodle: Using Learning Communities to Create an Open Source Course Management System / M. Dougiamas, P. C. Taylor // EdMedia+ Innovate Learning. — 2003. — С. 171-178. Association for the Advancement of Computing in Education (AACE).
15. Cole, J., & Foster, H. Using Moodle: Teaching with the Popular Open Source Course Management System / J. Cole, H. Foster. — O'Reilly Media, Inc., 2008.
16. Rice, W. Moodle E-Learning Course Development: A Complete Guide to Successful Learning Using Moodle / W. Rice. — Packt Publishing Ltd., 2006.
17. Martinez, M. Designing Learning Objects for Moodle / M. Martinez. — Packt Publishing Ltd., 2010.
18. Iftakhar, S. Google Classroom: What works and how? / S. Iftakhar // Journal of Education and Practice. — 2016. — № 7(2). — С. 12-19.
19. Beal, V. Google Classroom for Educators / V. Beal // Tech & Learning Magazine. — 2017.
20. Basilaia, G., & Kvavadze, D. Transition to Online Education in Schools during a SARS-CoV-2 Coronavirus (COVID-19) Pandemic in Georgia / G. Basilaia, D. Kvavadze // Pedagogical Research. — 2020. — № 5(4). — em0060.
21. Shaharanee, I. N. M., Jamil, J. M., & Rodzi, S. S. The Application of Google Classroom as a Tool for Teaching and Learning / I. N. M. Shaharanee, J. M. Jamil, S. S. Rodzi // Journal of Telecommunication, Electronic and Computer Engineering (JTEC). — 2016. — № 8(10). — С. 5-8.
22. Dabbagh, N., & Kitsantas, A. Personal Learning Environments, social media, and self-regulated learning: A natural formula for connecting formal and informal learning / N. Dabbagh, A. Kitsantas // The Internet and Higher Education. — 2012. — № 15(1). — С. 3-8.
23. Robinson, B. Integrating third-party tools in Canvas: A case study / B. Robinson // Journal of Educational Technology Systems. — 2016. — № 45(1). — С. 3-20.
24. Wang, J. Evaluating Canvas LMS for effective teaching and learning / J. Wang // Journal of Educational Computing Research. — 2014. — № 51(1). — С. 5-23.
25. Brown, S. Challenges and solutions for integrating Canvas LMS / S. Brown // Journal of Information Technology Education: Innovations in Practice. — 2018. — № 17. — С. 1-12.
26. Gamma, E., Helm, R., Johnson, R., & Vlissides, J. Design Patterns: Elements of Reusable Object-Oriented Software / E. Gamma, R. Helm, R. Johnson, J. Vlissides. — Addison-Wesley, 1994.
27. Booch, G. Object-Oriented Analysis and Design with Applications / G. Booch. — Addison-Wesley, 2007.
28. Rumbaugh, J., Jacobson, I., & Booch, G. The Unified Modeling Language Reference Manual / J. Rumbaugh, I. Jacobson, G. Booch. — Addison-Wesley, 2004.
29. Larman, C. Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and the Unified Process / C. Larman. — Prentice Hall, 2002.
30. Nielsen, J. Usability Engineering / J. Nielsen. — Morgan Kaufmann, 1994.
31. Krug, S. Don't Make Me Think, Revisited: A Common Sense Approach to Web Usability / S. Krug. — New Riders, 2014.
32. Shneiderman, B. Designing the User Interface: Strategies for Effective Human-Computer Interaction / B. Shneiderman. — Pearson, 2016.
33. Norman, D. The Design of Everyday Things / D. Norman. — Basic Books, 2013.
34. Практический опыт внедрения "1С:Университет ПРОФ" в АНО ВО "РосНОУ" (г. Москва). Результаты внедрения и перспективы использования программного продукта в вузе // Новые информационные технологии в образовании 2017 [электронный ресурс]: электрон. документ. Режим доступа: <https://educonf.1c.ru/conf2017/thesis/1267/>. — (Дата обращения: 05.06.2024).
35. Dougiamas, M., & Taylor, P. C. Moodle: Using Learning Communities to Create an Open Source Course Management System / M. Dougiamas, P. C. Taylor // EdMedia+ Innovate Learning. — 2003. — С. 171-178. Association for the Advancement of Computing in Education (AACE).
36. Dabbagh, N., & Kitsantas, A. Personal Learning Environments, social media, and self-regulated learning: A natural formula for connecting formal and informal learning / N. Dabbagh, A. Kitsantas // The Internet and Higher Education. — 2012. — № 15(1). — С. 3-8.