| Министерство образования и молодежной политики Свердловской области  Государственное автономное профессиональное образовательное учреждение  Свердловской области  «Камышловский педагогический колледж» |
| --- |

Дипломный проект

**Разработка автоматизированной информационной системы «Контроль состояния здоровья студентов ГАПОУ СО «Камышловский педагогический колледж» средствами VisualStudio C# и SQLServer**

специальность: 09.02.07 Информационные системы и программирование

**Исполнитель:**

Обоскалов Н.Д.,

студент 4П группы

**Руководитель:**

Мясников С.В.,

преподаватель

Камышлов, 2024

[**Пояснительная записка 2**](#_heading=h.2p2csry)

[Введение 3](#_heading=h.3o7alnk)

# **Пояснительная записка**

## Введение

В информационном обществе сегодня эффективность любой деятельности тесно связана с обработкой огромных объемов данных. Ручное управление информацией становится неэффективным и медленным. Внедрение передовых компьютерных технологий становится неотложной необходимостью. Системы хранения и обработки данных, включая СУБД, играют ключевую роль в оптимизации работы с информацией. Они обеспечивают хранение и быстрый доступ к данным, а также их эффективную обработку.

В медицинских кабинетах колледжей часто требуется заполнение, хранение и печать большого количества отчетов и документов. Это занимает много времени и усилий при ручной обработке на бумаге, что снижает эффективность работы сотрудников. В связи с этим автоматизация информационных систем становится необходимостью для оптимизации процессов хранения и обработки данных.

В исследовании темы дипломной работы использовались следующие нормативно-правовые акты: положение о выпускной квалификационной работе студентов; положение об индивидуальном проекте студентов; положение о курсовой работе (проекте) студентов. На основе вышеупомянутых актов, утверждённых в организации, удалось получить подробную информацию о хранении учебно-исследовательских работ: в каких форматах хранятся работа (проекты); срок хранения индивидуальных проектов, курсовых работ (проектов) и выпускных квалификационных работ; Федеральный закон от 27.07.2006 № 152-ФЗ «О персональных данных»; Федеральный закон от 23.12.2013 № 426-ФЗ «Об охране здоровья граждан в Российской Федерации»; Приказ Министерства образования и науки Российской Федерации от 28.12.2018 № 1739 «Об утверждении Порядка проведения медицинского освидетельствования обучающихся в образовательных организациях, реализующих образовательные программы начального профессионального образования, среднего профессионального образования, высшего образования».

На сегодняшний день медицинские данные хранятся в некомпьютерной системе, включающей журналы учета входящей и исходящей документации, острого и хронического заболевания, травм студентов, а также архив рабочей документации. Все процессы, включая обработку документов, составление отчетов, поиск информации и ведение статистики, выполняются вручную. Это требует значительных усилий и времени, особенно при повторном заполнении бланков с одинаковой информацией, например, фамилий студентов.

Вышеперечисленные проблемы могут быть решены с помощью автоматизированной информационной системы. Данные будут вводиться сразу при получении, а система автоматически заполнит соответствующие бланки документов. Готовые документы можно немедленно распечатать или сохранить в электронном виде.

Учитывая вышеизложенное, можно выделить ряд противоречий:

1. Между необходимостью применения информационных технологий в решении вопросов автоматизированной обработки информационных потоков и отсутствием необходимой автоматизированной информационной системы в организации;
2. Между существующими научными теориями, положениями, доказывающими необходимость использования АИС, и недостаточными ресурсами (материально техническими, кадровыми), обеспечивающими процесс внедрения автоматизированных информационных систем;
3. Между недостаточной эффективностью сбора сведений о медицинских данных студентов и отсутствием автоматизированной информационной системы в организации.

Выявленные противоречия указывают на необходимость внедрения информационных технологий и программного обеспечения в управлении медицинскими данными и контроле здоровья студентов для повышения эффективности, и безопасности процессов в колледже.

Выявленные противоречия позволили определить проблему: необходимость автоматизации процесса контроля состояния здоровья студентов колледжа средствами автоматизированных информационных систем (АИС).

Важность и актуальность рассматриваемой проблемы, ее недостаточная разработанность определили выбор темы: «Разработка автоматизированной информационной системы «Контроль состояния здоровья студентов ГАПОУ СО «Камышловский педагогический колледж»» средствами Visual Studio C# и SQL Server»

Следовательно, объектом проектирования является процесс контроля состояния здоровья студентов в ГАПОУ СО «Камышловский педагогический колледж», включающий в себя сбор и обновление медицинских данных, проведение профилактических мероприятий, реагирование на медицинские проблемы, запрос и обработку медицинских документов, формирование отчетности и сотрудничество с медицинскими организациями.

На основе противоречий и проблемы была сформирована цель выпускной квалификационной работы: проектирование информационной системы «Контроль состояния здоровья студентов ГАПОУ СО «Камышловский педагогический колледж».

Для достижения поставленной цели необходимо решить следующие задачи:

1. Выполнить описание предметной области, выбрать и проанализировать методы решения;
2. Выполнить проектирование базы данных, осуществить проектирование приложений конечных пользователей, а именно выбрать среду разработки приложения, выполнить проектирование с использованием UML диаграмм и методологии SADT, спроектировать пользовательский интерфейс;
3. Реализовать информационную систему на программном уровне;
4. Выполнить тестирование приложений, провести отладку программных модулей и разработать сопровождающую техническую документацию;
5. Провести оценку экономической эффективности.

Для решения поставленных задач применялся комплекс взаимодополняющих теоретических и эмпирических методов исследования. К методам теоретического исследования, использованным в работе, можно отнести теоретический анализ научной и методической литературы, анализ профессионального опыта, обобщение, описание. Эти методы направлены на создание теоретических обобщений, установление и формулирование закономерностей изучаемого явления. К эмпирическим методам, использованным в работе, можно отнести тестирование, метод описания, метод бесед.

Для проектирования и разработки информационной системы использовалось структурно-функциональная методология. В структурном анализе и проектировании используются различные модели, описывающие: функциональную структуру системы; последовательность выполняемых действий; передачу информации между функциональными процессами; отношения между данными.

Я использовал следующие распространённые модели:

* DFD (Data Flow Diagrams) – диаграммы потоков данных;
* UML (Unified Modeling Language) – язык графического описания;
* Функциональная модель SADT
* Диаграмма «Сущность - связь»

Новизна дипломной работы заключается в разработке АИС, специально адаптированной к потребностям образовательного учреждения, что учитывает специфические требования и особенности процессов контроля состояния здоровья студентов. Интеграции современных методов и технологий для эффективной обработки, и анализа медицинских данных, что позволяет повысить точность, доступность и безопасность информации.

Практическая значимость курсового проекта состоит в возможности проектирования информационной системы, которая позволит разгрузить сотрудника медицинского кабинета и использовать современные цифровые технологии для контроля состояния здоровья студентов. Исследование может представлять интерес для студентов, обучающихся по специальности 09.02.07 «информационные системы и программирование».

Данная выпускная квалификационная работа состоит из основных разделов: пояснительная записка дипломного проекта, которая содержит введение, описание предметной области, анализ и выбор методов решения задач, проектирование информационной системы, программная реализация информационной системы, оценка экономической эффективности проекта, заключение; список литературы, диск с программным продуктом, технической документацией и демонстрационным материалом, приложения.

В первом разделе проводится анализ и описание предметной области, и выбор методов решения, моделируя потоки данных, определяются требования будущих пользователей, целей, задач и функций программного продукта, выполняется анализ существующих аналогов решения проекта и разработка технического задания.

Во втором разделе проводится проектирование информационной системы, в которое входит проектирование базы данных и приложений конечных пользователей. Проектирование базы данных представляет из себя создание логической модели, представление глоссариев и спецификаций, создание физической модели базы данных. Проектирование приложений конечных пользователей включает в себя выбор среды разработки приложения для АИС, проектирование с использованием UML диаграмм, определение функций программного продукта с применением методологии SADT и проектирование пользовательского интерфейса.

В третьем разделе приводится программная реализация информационной системы, в которую входит интерфейс приложения, функции поиска и фильтрации данных, функции просмотра и генерации отчётов, а также дополнительные функции приложения. Помимо вышеперечисленных подразделов описывается тестирование приложения и отладка программных модулей. В четвёртом разделе описывается разработка сопровождающей технической документации, а именно руководство пользователя.

В пятом разделе приводится обоснование методики расчета социальной эффективности результатов дипломного проектирования, расчет показателей социальной эффективности.

**Описание предметной области, анализ и выбор методов решения**

**Исследование предметной области**

Исследование — это процесс познания определенной предметной области, объекта или явления с определенной целью.

Процесс исследования заключается в наблюдении свойств объектов с целью выявления и оценки важных, с точки зрения субъекта-исследователя, закономерных отношений между показателями данных свойств.

Решение любой задачи в сфере разработки программного обеспечения должно начинаться с изучения предметной области.

Предметная область — это мысленно ограниченная область реальной действительности, подлежащая описанию или моделированию и исследованию

Анализ предметной области, позволяет выделить её сущности, определить границы проекта и первоначальные требования к функциональности. Модель предметной области должна быть документирована, храниться и поддерживаться в актуальном состоянии до этапа реализации. Для документирования могут быть использованы различные средства.

Описание предметной области:

В медицинском кабинете ГАПОУ СО «Камышловский педагогический колледж» происходит контроль состояния здоровья студентов. Общей целью данной предметной области является обеспечение безопасности и заботы о здоровье студентов, а также оперативное выявление и управление медицинскими проблемами, способными повлиять на их обучение и общее благополучие.

Предметная область контроля состояния здоровья студентов охватывает процессы мониторинга, учета и обеспечения безопасности здоровья студентов в образовательном учреждении. Эта область включает в себя сбор и анализ информации о физическом состоянии студентов, хронических заболеваниях, аллергиях и других медицинских параметрах.

Основные составляющие:

1. Процесс отслеживания состояния здоровья включает в себя регулярные обновления медицинских данных, проведение профилактических мероприятий и обеспечение своевременного реагирования на потенциальные медицинские проблемы студентов.

2. Система контроля состояния здоровья в учебном заведении также предусматривает запросы медицинских документов, в том числе устные запросы у студентов или ответственных за посещаемость, а также официальные запросы у медицинских организаций и предыдущих учебных заведений.

3. Отчетность о пропусках по болезни становится неотъемлемой частью этой предметной области, где студенты обязаны предоставлять медицинские справки ответственным за учет посещаемости, а эти данные впоследствии анализируются медицинским работником для дальнейшего контроля.

4. Сотрудничество с медицинскими организациями становится важным аспектом, особенно при наблюдении или обследовании студентов во внешних медицинских учреждениях, где обеспечивается доступ к соответствующим медицинским документам для более надежного мониторинга состояния здоровья студентов.

Информационная модель — модель объекта, представленная в виде информации, описывающей существенные для данного рассмотрения параметры и переменные величины объекта, связи между ними, входы и выходы объекта и позволяющая путём подачи на модель информации об изменениях входных величин моделировать возможные состояния объекта.

Описание информационной модели:

Медицинский работник в устной форме запрашивает документы о состоянии здоровья у студента или ответственного за посещаемость, по электронной почте или запросом в виде документа у медицинской организации или прошлого учебного заведения. Ответственный за посещаемость сдает в виде документа справку о пропуске. Справку о пропуске по болезни в виде документа студент сдает ответственному за посещаемость. Студент наблюдается или наблюдался в медицинской организации что подтверждается документами (см. Рисунок 1).

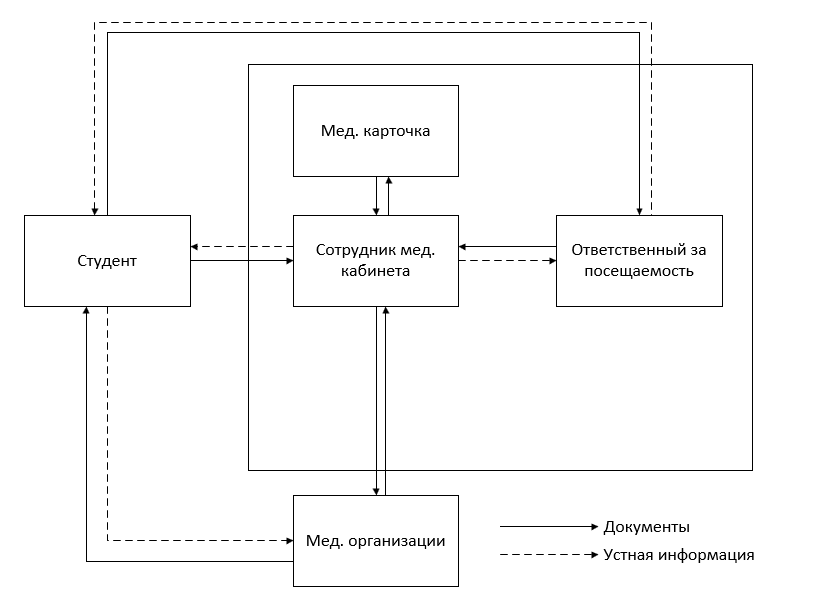


Рисунок 1 - Информационная модель

После построения информационной модели была построена диаграмма вариантов использования (use case diagram) в UML (Unified Modeling Language), которая отражает отношения между акторами и прецедентами. Данная диаграмма является составной частью модели прецедентов, позволяющая описать систему на концептуальном уровне. Она состоит из графической диаграммы, описывающей действующие лица и прецеденты, а также спецификации, представляющего собой текстовое описание конкретных последовательностей действий, которые выполняет пользователь при работе с системой. Кроме того, диаграмма достаточно проста, чтобы ее мог понять заказчик, следовательно, её можно использовать для согласования ТЗ, ведь диаграмма описывает функциональные требования к системе.

В качестве подсистемы для построения диаграммы вариантов использования была выбрана система «Контроль состояния здоровья студентов колледжа». Далее были описаны основные объекты: актёры, прецеденты и связи между ними (см. Рисунок 2).

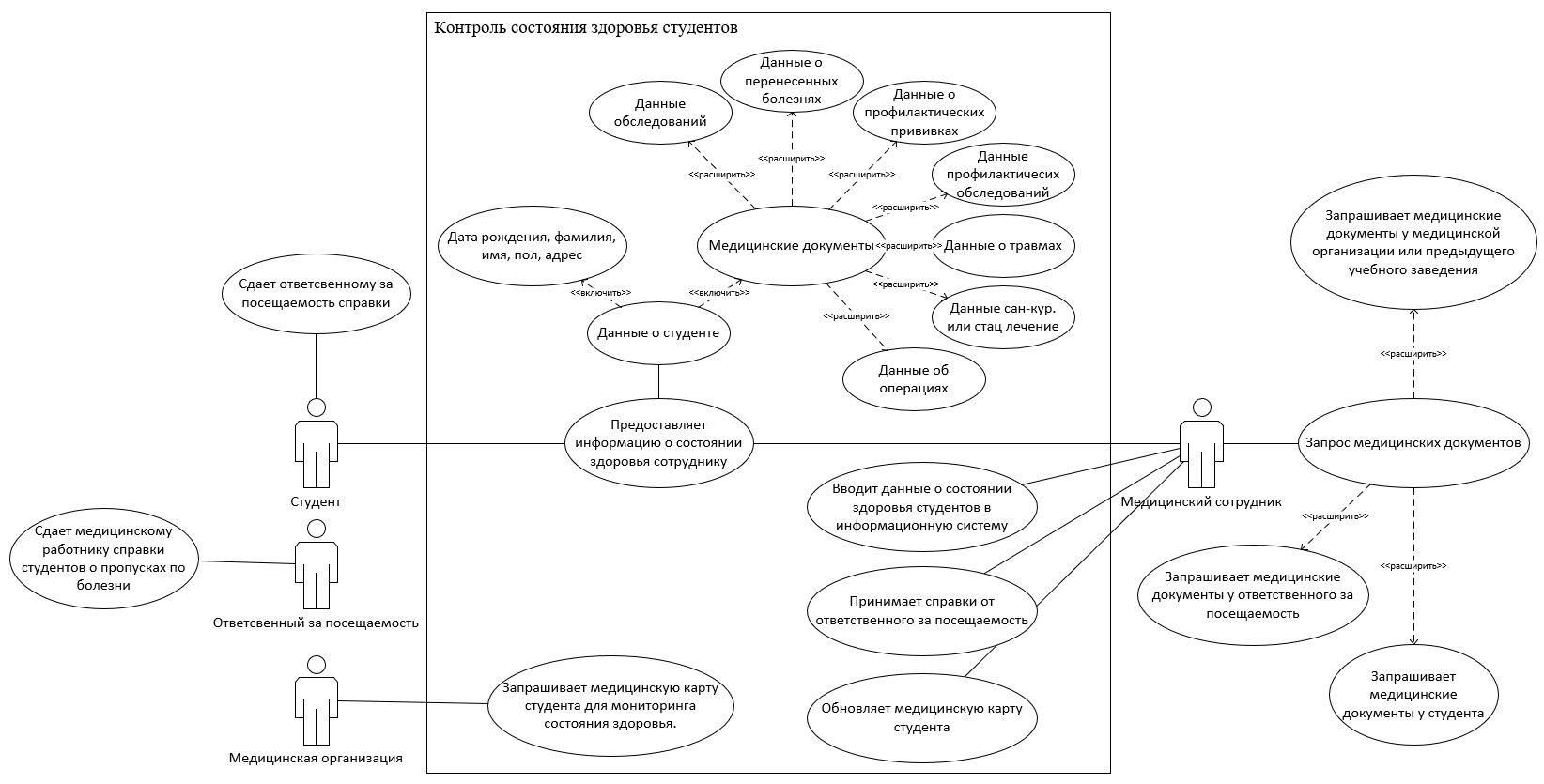


Рисунок 2 - Диаграмма вариантов использования

Описание диаграммы вариантов использования:

Основные группы пользователей системы:

1. Медицинский работник колледжа. Основной пользователь системы, ответственный за мониторинг состояния здоровья студентов и ведение медицинских данных.
2. Студент. Пользователь, чье здоровье отслеживается в системе. Предоставляет медицинские данные и документы.
3. Ответственный за посещаемость в группе. Пользователь, отвечающий за учет посещаемости студентов в группе и сдачу медицинских справок.
4. Медицинская организация. Внешний пользователь, имеющий интерес к медицинским данным студентов.

Варианты использования:

1. Отслеживание состояния здоровья студентов.

Описание: Медицинский работник вносит и обновляет данные о состоянии здоровья студентов в информационной системе.

Связи: Связан с актерами "Медицинский работник колледжа" и "Студент".

1. Запрос медицинских документов.

Описание: Медицинский работник запрашивает медицинские документы у студентов, ответственного за посещаемость и медицинских организаций.

Связи: Связан с актерами "Медицинский работник колледжа", "Студент", "Ответственный за посещаемость в группе" и "Медицинская организация".

1. Отчетность о пропусках по болезни.

Описание: Ответственный за посещаемость сдает медицинскому работнику справки о пропусках занятий по болезни студентов.

Связи: Связан с актерами "Ответственный за посещаемость в группе" и "Медицинский работник колледжа".

1. Информационная система.

Описание: Система для ввода, хранения и обновления данных о состоянии здоровья студентов.

Связи: Связан с актером "Медицинский работник колледжа".

1. Сотрудничество с медицинскими организациями.

Описание: Медицинский работник обеспечивает сотрудничество с внешними медицинскими организациями для обмена информацией о состоянии здоровья студентов.

Связи: Связан с актерами "Медицинский работник колледжа" и "Медицинская организация".

Отношения между актерами и прецедентами:

1. Медицинский работник колледжа. Взаимодействует с прецедентами "Отслеживание состояния здоровья студентов", "Запрос медицинских документов", "Отчетность о пропусках по болезни", "Информационная система" и "Сотрудничество с медицинскими организациями".
2. Студент. Взаимодействует с прецедентами "Отслеживание состояния здоровья студентов" и "Запрос медицинских документов".
3. Ответственный за посещаемость в группе. Взаимодействует с прецедентом "Отчетность о пропусках по болезни".
4. Медицинская организация. Взаимодействует с прецедентами "Запрос медицинских документов" и "Сотрудничество с медицинскими организациями".

**Моделирование потоков данных (выделение бизнес-процессов)**

DFD (Диаграмма Потоков Данных) позволяет визуализировать потоки данных в системе и их обработку, идентифицируя основные процессы, хранилища данных и внешние сущности. Это упрощает анализ и проектирование систем, фокусируя внимание на передаче, обработке и хранении информации внутри системы.

Бизнес-процесс: формирование медицинской карточки в медицинском кабинете колледжа

1. Инициирование процесса. Мед. комиссия инициирует процесс, запрашивая отчёт у сотрудника медицинского кабинета в колледже.
2. Запрос на получение отчёта. Медицинская комиссия запрашивает у сотрудника отчёт.
3. Требование на составление отчёта. Передача запроса на получение отчёта сотруднику медицинского кабинета.
4. Ручное составление отчёта. Сотрудник медицинского кабинета вручную составляет отчёт для медицинской комиссии.
5. Готовый отчёт. Передача готового отчёта медицинской комиссии.

Чтобы построить модель после реинжиниринга провел оценку по количеству критических факторов успеха (КФУ). Выбрал факторы для оценки: скорость, качество, стоимость, удобство обслуживания. Далее по каждому бизнес-процессу поставил оценку выполнения. Сосчитал средний бал по формуле. Представил результаты (см. Таблицу 1) оценки бизнес-процессов по критическим факторам (КФУ).

Таблица 1 - Сопоставление бизнес-процессов с критическими факторами успеха

| Бизнес- процесс/критерии | Скорость | Качество | Удобство обслуживания | Средний балл процесса |
| --- | --- | --- | --- | --- |
| Запрос на получение отчёта | 2 | 4 | 2 | 2,6 |
| Вес | 0,4 | 0,3 | 0,4 | 1 |
| Ручное составление отчёта | 3 | 3 | 2 | 2,6 |
| Вес | 0,4 | 0,3 | 0,4 | 1 |
| Общая оценка | | | | 2,6 |

Бизнес-процесс: формирование медицинской карточки в медицинском кабинете колледжа после реинжиниринга

1. Инициирование процесса. Мед. комиссия инициирует процесс, запрашивая отчёт у сотрудника медицинского кабинета в колледже.
2. Формирование отчёта с помощью программы. Сотрудник медицинского кабинета получает запрос на формирование отчёт от медицинской комиссии и с помощью программы удобно и быстро создаёт отчёт.
3. Готовый отчёт. Передача готового отчёта медицинской комиссии.

Построил таблицу КФУ после реинжиниринга (см. Таблицу 2).

Таблица 2 - Сопоставление бизнес-процессов с критическими факторами успеха после реинжиниринга

| Бизнес- процесс/критерии | Скорость | Качество | Удобство обслуживания | Средний балл процесса |
| --- | --- | --- | --- | --- |
| Запрос на получение отчёта | 4 | 5 | 5 | 4,6 |
| Вес | 0,4 | 0,3 | 0,4 | 1 |
| Ручное составление отчёта | 4 | 5 | 5 | 4,6 |
| Вес | 0,4 | 0,3 | 0,4 | 1 |
| Общая оценка | | | | 4,6 |

**Определение требований будущих пользователей, целей, задач и функций программного продукта**

Для определения требований будущих пользователей и функций программного продукта была построена SADT модель.  
SADT (Structured Analysis and Design Technique) используется для моделирования структуры и функций программного продукта. В случае автоматизированной информационной системы «Контроль состояния здоровья студентов ГАПОУ СО «Камышловский педагогический колледж»», применение SADT может быть представлено следующим образом:

Входящие данные:

* Данные о студенте (ФИО, дата рождения, пол, адрес, общежитие, группа)
* Медицинская история студента (хронические заболевания, аллергии)
* Медицинские данные с предыдущего места обучения, медицинской организации
* Справки о пропусках

Выходные данные:

* Медицинские отчеты
* Уведомления студентов о результатах осмотров и необходимости прохождения вакцинаций

Управление:

* Роль пользователя
* Список медицинских документов
* Данные о пройденных медицинских осмотрах, вакцинациях и т.д.

Механизм:

* Сотрудник
* Ответственный за пропуски

Из SADT диаграммы SADT A0 можно определить следующие требования пользователей:

* Медицинский персонал должен иметь возможность быстрого и удобного доступа к медицинским данным студентов; функционал для ввода, редактирования и удаления медицинской информации; возможность генерации отчетов о состоянии здоровья студентов и пропусках по болезни; доступ к истории медицинских данных студентов для анализа и управления.
* Студент должен иметь возможность предоставления медицинских справок и документов; доступ к своим медицинским данным и истории посещений медицинского кабинета; возможность просмотра графика проведения профилактических мероприятий.
* Пользователи должны иметь возможность вводить данные о студенте, включая ФИО, дату рождения, пол, адрес, общежитие, группу.
* Пользователи должны иметь возможность вводить медицинскую историю студента, включая хронические заболевания и аллергии.
* Пользователи должны иметь возможность вводить медицинские данные с предыдущего места обучения или медицинской организации.
* Пользователи должны иметь возможность вводить справки о пропусках.
* Пользователи должны иметь возможность получать медицинские отчеты.
* Пользователи должны получать уведомления о результатах осмотров и необходимости прохождения вакцинаций.
* Пользователи должны иметь возможность управлять ролями пользователей.
* Пользователи должны иметь возможность просматривать список медицинских документов.
* Пользователи должны иметь возможность просматривать данные о пройденных медицинских осмотрах, вакцинациях и т.д.
* Пользователи должны иметь возможность назначать ответственного за пропуски.
* Пользователи должны иметь возможность назначать сотрудника, ответственного за работу с системой.

Цели программного продукта:

* Оптимизация процессов учета и контроля состояния здоровья студентов: обеспечение централизованного доступа к медицинской информации, сокращение времени на административные процедуры и повышение эффективности работы медицинского персонала.
* Повышение качества медицинского обслуживания: предоставление инструментов для анализа и мониторинга состояния здоровья студентов, что позволит оперативно выявлять проблемы и предоставлять качественную медицинскую помощь.

Задачи программного продукта:

* Создание базы данных медицинских записей студентов: Разработка функционала для хранения и обработки информации о физическом состоянии, хронических заболеваниях, аллергиях и других медицинских параметрах студентов.
* Разработка интерфейса пользователя: Создание удобного и интуитивно понятного пользовательского интерфейса для всех категорий пользователей с учетом их потребностей и привычек.
* Реализация функционала формирования отчетов и аналитики: Разработка возможности генерации отчетов о состоянии здоровья студентов, пропусках по болезни, а также инструментов для анализа медицинских данных.

Функции программного продукта:

* Учет медицинских данных студентов: Ввод, хранение и обновление информации о состоянии здоровья студентов, включая результаты осмотров, анализов и консультаций.
* Генерация отчетов и аналитика: Формирование различных отчетов о состоянии здоровья

**Анализ существующих аналогов решения проекта**

В ходе разработки создания приложения, были проанализированы 2 аналога приложения: MEDESK и eClinic. Проанализировав существующие аналоги разрабатываемой ИС, описал основные возможности, написал их плюсы и минусы в итоге сделал вывод.

* 1. MEDESK

Облачная информационная система, предназначенная для учета, управления и анализа медицинских данных врачей и пациентов (см. Рисунок 3). Создана для удобного доступа к медицинской информации в любое время и место.

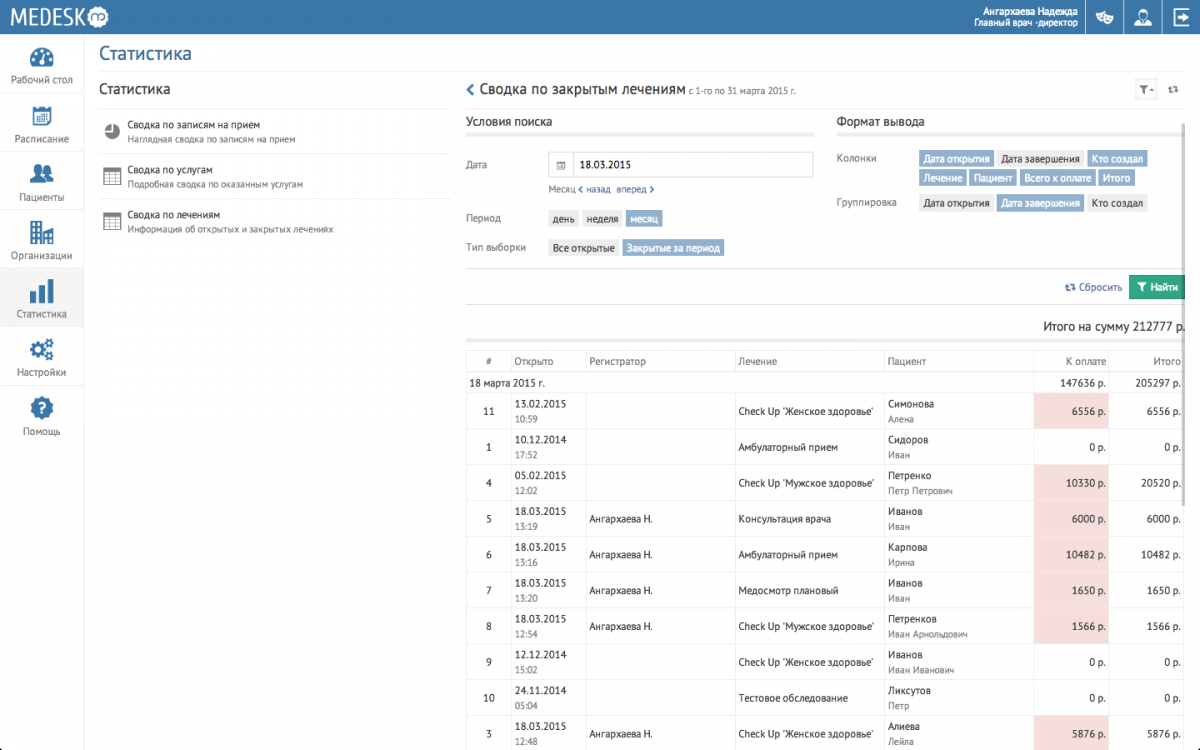


Рисунок 3 - Программа MEDESK

Основные возможности:

* Ведение электронной медицинской карты пациента.
* Учет и анализ медицинских данных и результатов лабораторных исследований.

Плюсы:

* Удобный интерфейс
* Высокий уровень безопасности данных
* Возможность работы через веб-интерфейс или мобильное приложение

Минусы:

* Дорогой тарифный план
* Не всегда удобный поиск и доступ к необходимой информации в больших базах данных

Лицензия: платная

* 1. eClinic

Программа позволяющая управлять графиком работы врачей, записывать пациентов на прием, информировать клиентов о записях - система отправит sms талон и напоминание о предстоящем приеме (см. Рисунок 4). Расширять список клиентов за счет записи с сайта клиники.

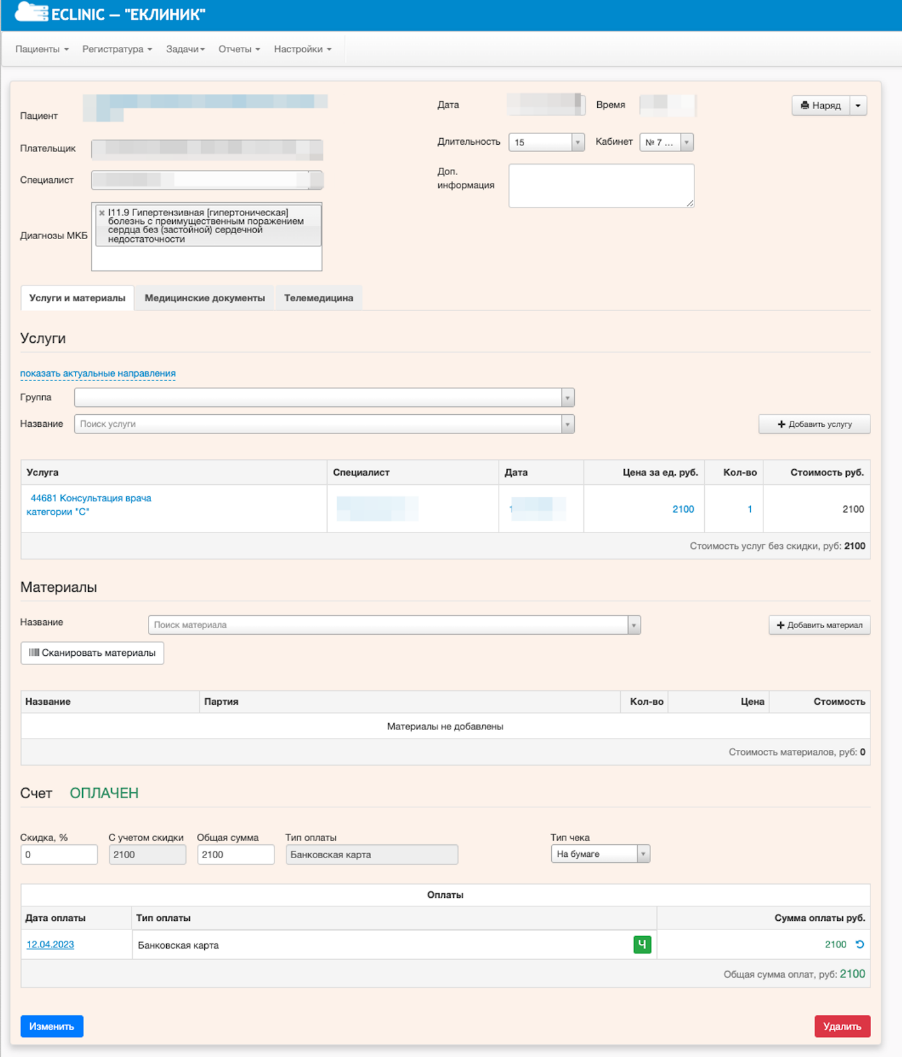


Рисунок 4 - Интерфейс ECLINIC

Основные возможности:

* Учет пациентов и их общей медицинской истории
* Учет всех клинических документов пациентов и документооборота

Плюсы:

* Понятный интерфейс со сложной организацией данных
* Подключение государственных центров обмена медицинской информации

Минусы:

* Высокая стоимость системы
* Ограниченные варианты графических отчетов
* Не всегда удобный интерфейс

Лицензия: платная

Вывод:

Разрабатываемая информационная система предоставляет значительные преимущества по сравнению с аналогами благодаря гарантированной безопасности данных, удобству использования и повышенной эффективности работы пользователей. Глубокая ответственность за защиту информации обеспечивает соблюдение высоких стандартов безопасности, в то время как индивидуальная настройка под требования заказчика делает систему легкой в освоении и интуитивно понятной для персонала. Это приводит к повышенной производительности и эффективности в рамках рабочих процессов, сделав разрабатываемую ИС превосходным выбором для улучшения бизнеса и обслуживания клиентов.

**Разработка технического задания**

Техническое задание – исходный документ на проектирование технического объекта (изделия).

Техническое задание оформляют в соответствии с ГОСТ 19.201-78 Единая система программной документации и ГОСТ 24.701-86. Единая система стандартов автоматизированных систем управления. Надежность автоматизированных систем управления.

Техническое задание устанавливает:

1. Основное назначение разрабатываемого объекта;
2. Его технические характеристики;
3. Показатели качества и технико-экономические требования;
4. Предписание по выполнению необходимых стадий создания документации (конструкторской, технологической, программной и т. д.) и её состав;
5. Специальные требования.

Техническое задание содержит следующие разделы:

1. Введение;
2. Основания для разработки;
3. Назначение разработки;
4. Требования к программе или программному изделию;
5. Требования к программной документации;
6. Технико-экономические показатели;
7. Стадии и этапы разработки;
8. Порядок контроля и приемки;
9. В техническое задание допускается включать приложения.

При разработке технического задания были использованы ГОСТы, содержащие информацию по разработке технического задания для информационной системы, такие как ГОСТ 19.104-78 Единая система программной документации (ЕСПД). Основные надписи (с Изменением N 1) от 18 декабря 1978, ГОСТ 19.106-78 Единая система программной документации (ЕСПД). Требования к программным документам, выполненным печатным способом (с Изменением N 1) от 18 декабря 1978, ГОСТ 19.105-78 Единая система программной документации (ЕСПД) [12]. Общие требования к программным документам (с Изменением N 1) от 18 декабря 1978, ГОСТ 2.301-68 Единая система конструкторской документации (ЕСКД). Форматы (с Изменениями N 1, 2, 3) от 28 мая 1968 и ГОСТ 19.105-78 Единая система программной документации (ЕСПД). Общие требования к программным документам (с Изменением N 1) от 18 декабря 1978.

**Проектирование информационной системы**

**Проектирование базы данных**

Проектирование базы данных представляет собой важный этап, включающий создание комплекса взаимосвязанных моделей данных, где каждая модель отражает определенные аспекты предметной области. Этот процесс тесно связан с обработкой и манипулированием данными, поскольку целью базы данных является хранение, организация и эффективное использование информации.

Проектирование базы данных включает две основные фазы:

1. Логическое моделирование. В этой фазе разработчик собирает требования к базе данных, анализирует предметную область и создает абстрактную модель данных, не привязанную к конкретной системе управления базами данных (СУБД). Логическая модель описывает сущности, их атрибуты и взаимосвязи между ними.
2. Физическое моделирование. На этой стадии разработчик преобразует логическую модель в конкретную структуру данных, оптимизированную для выбранной СУБД и потребностей конкретных приложений пользователей. Это включает определение таблиц, индексов, ключей, хранимых процедур и других элементов, необходимых для реализации функциональности базы данных.

**Создание логической модели**

С помощью Microsoft Visio я создал логическую модель базы данных, используя инструменты и возможности программы для визуального представления структуры данных. Для этого я определил сущности и их атрибуты, установил связи между ними и настроил отношения между сущностями. Затем я создал таблицы и столбцы на основе определенных сущностей и атрибутов, а также сформировал SQL код для создания таблиц, индексов и других объектов базы данных на основе созданной логической модели. Это позволило мне лучше понять структуру данных и их взаимосвязи, а также упростило процесс создания и поддержки базы данных.

Создание логической модели данных является одним из главных средств моделирования данных, так как позволяет взглянуть на данные, основанных и соответствующих реальным.

Моделирование данных осуществляется с целью обеспечения разработчика информационной системы концептуальной схемы базы данных в форме одной или нескольких локальных моделей, которые относительно легко могут быть отображены СУБД.

Наиболее распространенным средством моделирования данных являются диаграммы сущность-связь – ERD (Entity Relationship Diagram). С их помощью определяются важные для предметной области объекты, их свойства и отношения друг с другом.

На каждом этапе проектирования информационной модели будущей системы разрабатывается соответствующая модель данных: логическая и физическая модели.

Логический уровень представления модели данных - это уровень абстракции, на котором описываются сущности, их атрибуты и отношения между ними, не привязываясь к конкретной СУБД или физической реализации. Этот уровень представления позволяет разработчику создавать концептуальную модель данных, которая отражает требования бизнеса и правила предметной области.

На логическом уровне представления модели данных используются следующие элементы:

* Сущности (entities) - объекты или концепции предметной области, которые должны быть представлены в базе данных.
* Атрибуты (attributes) - свойства или характеристики сущностей, которые описывают их состояние или поведение.
* Отношения (relationships) - связи между сущностями, которые определяют, как они взаимодействуют друг с другом.

Различают три уровня логической модели, которые отличаются по глубине представления информации в ней.

* ER - уровень - Диграмма сущность-связь (Entity Relationship Diagram);
* KB - уровень - Модель данных, основанная на ключах (Key Based model);
* FA - уровень - Полная атрибутивная модель (Fully Attributed model).

Методология моделирования IDEF1X представляет собой расширенный стандарт IDEF1, предназначенный для описания данных (информации). Она основана на языке семантического моделирования, использующем концепцию "сущность — связь", которая позволяет определять данные и связи между ними. Данная методология применяется для создания информационной модели предметной области путем идентификации ее сущностей и связей между ними. Часто она используется для описания данных с целью последующей автоматизации их обработки с помощью систем управления базами данных. Таким образом, можно сказать, что модели данных в нотации IDEF1X используются для создания баз данных.

Диаграмма "сущность — связь" используется для описания данных, на верхнем уровне в целях визуализации данных для руководителей и экспертов, не обладающих специальными знаниями в области моделирования. Она показывает основные сущности и связи между ними.

Модель данных, основанная на ключах, предназначена для более детального описания данных. В ней показываются ключевые атрибуты, по которым идентифицируются сущности и связываются друг с другом.

Полная атрибутивная модель дает детальное представление структуры данных. Она позволяет представить данные в третьей нормальной форме и включает все сущности, их атрибуты и связи.

Нормализация данных представляет собой процесс проверки и реорганизации сущностей и атрибутов в целях удовлетворения требований к реляционной модели данных. Нормализация позволяет быть уверенным, что каждый атрибут определен для своей сущности, дает возможность значительно сократить объем памяти для хранения информации и устранить аномалии в организации хранения данных. В результате проведения нормализации должна быть создана структура данных, при которой информация о каждом факте хранится только в одном месте. Процесс нормализации сводится к последовательному приведению структуры данных к нормальным формам — формализованным требованиям к организации данных. Известны шесть нормальных форм:

* первая нормальная форма (1NF):
* вторая нормальная форма (2NF);
* третья нормальная форма (3NF):
* нормальная форма Бойса — Кодда (усиленная 3NF);
* четвертая нормальная форма (4NF):
* пятая нормальная форма (5NF).

Основными элементами модели IDEF1X являются: сущности, атрибуты и отношения.

Сущность — это множество реальных или абстрактных объектов, обладающих общими характеристиками, например, персонал, явления, предметы. Каждый такой объект может быть представлен только одной уникальной сущностью.

Сущность имеет уникальное имя. Как правило, ее название обозначается существительным в единственном числе, изображается в виде прямоугольника.

Сущность имеет один или несколько экземпляров. Все экземпляры сущности уникальны, т.е. имеют различные характеристики. Таким образом, сущность имеет один или несколько атрибутов, которые ей принадлежат или наследуются в результате отношений с другими сущностями.

Поскольку все сущности неповторимы, то они должны иметь свой уникальный идентификатор. В качестве такого идентификатора выступают один или совокупность нескольких атрибутов, которые называются первичным ключом. Атрибут, идентифицирующий экземпляр сущности, называется простым ключом. Идентификатор, который состоит из нескольких атрибутов, называется составным ключом.

Первичный ключ, который наследуется от родительской сущности, называется внешним ключом.

Также есть ключи, по которым связываются сущности друг с другом. Он называется внешним ключом.

Отношения - это связи между двумя и более сущностями.

Сущность, обладающая некоторыми атрибутами, является типом информации, которая хранится в базе данных. Связь этих типов информации друг с другом показывается в виде отношений между сущностями. В рамках методологии IDEF1X существуют следующие типы отношений:

* идентифицирующие отношения (сущность однозначно определяет другую сущность);
* не идентифицирующие отношения;
* отношения "многие ко многим";
* отношения категоризации.

В идентифицирующих отношениях дочерняя сущность всегда является зависимой сущностью от идентифицирующей.

Не идентифицирующие отношения используются в случаях, когда одна сущность связана с другой, но однозначно её не определяет. При данной связи первичный ключ одной сущности наследуется другой, но уже является не ключевым атрибутом.

Связь типа "многие ко многим" позволяет показать, что любой экземпляр сущности "Л" может иметь связь с несколькими экземплярами другой сущности "В", и наоборот.

Отношения категоризации подразумевают связь, при которой каждый экземпляр одной сущности связан с одним экземпляром категориальной сущности. А атрибуты первичного ключа категориальной сущности должны совпадать с атрибутами первичного ключа общей сущности.

Название отношений обозначается фразой, которая описывает отношения между сущностями.

Используя методологию IDEF1X, удалось спроектировать логическую модель FA-уровня.

Согласно построенной логической модели, были выделены сущности:

* 1. Сущность «USER» имеет 12 атрибутов: ID\_USER, ID\_USER\_ROLE, Surname, Name, Patronymic, DateBirth, HomeAddress, Hostel, ID\_GROUP, DateEnrollment, Color, ImageName.
  2. Сущность «PASS\_CONROLLER» имеет 2 атрибута: ID\_PASS\_CONROLLER, ID\_USER.
  3. Сущность «LOGIN\_PASSWORD» имеет 3 атрибута: Login, Password, ID\_USER.
  4. Сущность «OPERATIONS» имеет 3 атрибута:

ID\_SANITARY\_SPA\_TREATMENT, ID\_USER, Description.

* 1. Сущность «SANITARY\_SPA\_TREATMENT» имеет 3 атрибута: ID\_OPERATIONS, ID\_USER, Operations.
  2. Сущность «CHECK\_VACCINE» имеет 8 атрибута: ID\_CHECK\_VACCINE, ID\_USER, Date, BodyTemp, ResultInspecion, Vaccine, Permission, StateAfterThirtyMin.
  3. Сущность «HOSPITAL\_TREATMENT» имеет 3 атрибута: ID\_HOSPITAL\_TREATMENT, ID\_USER, Description.
  4. Сущность «USER\_ROLE» имеет 2 атрибута: ID\_USER\_ROLE, NameRole.
  5. Сущность «GROUP» имеет 3 атрибута: ID\_GROUP, GroupName, NumberPeople.
  6. Сущность «PASSES\_FOR\_ILLNESS» имеет 4 атрибута: ID\_PASSES\_FOR\_ILLNESS, ID\_USER, Date, Diagnosis.
  7. Сущность «PAST\_INFECTIOUS\_DIASES» имеет 4 атрибута: ID\_PAST\_INFECTIOUS\_DIASES, ID\_USER, Date, Description.
  8. Сущность «OPERATIONS» имеет 3 атрибута: ID\_ OPERATIONS, ID\_USER, Operations.
  9. Сущность «TRAUMA» имеет 3 атрибута: ID\_TRAUMA, ID\_USER, Description.
  10. Сущность «DATA\_MEDICAL\_CHECKUPS» имеет 6 атрибутов: ID\_DATA\_MEDICAL\_CHECKUPS, ID\_USER, ID\_MEDICAL\_ORGANIZATION, Year, Survey, SurveyResult.
  11. Сущность «MEDICAL\_ORGANIZATION» имеет 3 атрибута: ID\_MEDICAL\_ORGANIZATION, Name, Address.
  12. Сущность «SURVEYS» имеет 6 атрибутов: ID\_SURVEYS, ID\_USER, Year, Survey, SurveyResult, ID\_MEDICAL\_ORGANIZATION.

После создания сущностей определила, какие зависимые, а какие нет.

Зависимая таблица - это таблица, которая ссылается на первичный ключ другой таблицы с помощью внешнего ключа. Зависимая таблица зависит от существования записи в связанной таблице, на которую она ссылается. В зависимой таблице могут существовать ограничения целостности, такие как CASCADE, SET NULL или SET DEFAULT, которые определяют, что происходит с записями в зависимой таблице, когда связанная запись в другой таблице обновляется или удаляется. Зависимыми сущностями являются:

* Сущность «USER»: Зависит от «GROUP» через внешний ключ «ID\_GROUP» и от «USER\_ROLE» через внешний ключ «ID\_USER\_ROLE»
* Сущность «CHECK\_VACCINE»: Зависит от «USER» через внешний ключ «ID\_USER»
* Сущность «DATA\_MEDICAL\_CHECKUPS»: Зависит от «USER» через внешний ключ «ID\_USER» и от «MEDICAL\_ORGANIZATION» через внешний ключ «ID\_MEDICAL\_ORGANIZATION»
* Сущность «HOSPITAL\_TREATMENT»: Зависит от «USER» через внешний ключ «ID\_USER»
* Сущность «LOGIN\_PASSWORD»: Зависит от «USER» через внешний ключ «ID\_USER»
* Сущность «OPERATIONS»: Зависит от «USER» через внешний ключ «ID\_USER»;
* Сущность «PASS\_CONROLLER»: Зависит от «USER» через внешний ключ «ID\_USER»
* Сущность «PASSES\_FOR\_ILLNESS»: Зависит от «USER» через внешний ключ «ID\_USER»
* Сущность «PAST\_INFECTIOUS\_DIASES»: Зависит от «USER» через внешний ключ «ID\_USER»
* Сущность «PREVENTIVE\_VACCINE»: Зависит от «USER» через внешний ключ «ID\_USER» и от «MEDICAL\_ORGANIZATION» через внешний ключ «ID\_MEDICAL\_ORGANIZATION»
* Сущность «SANITARY\_SPA\_TREATMENT»: Зависит от «USER» через внешний ключ «ID\_USER»
* Сущность «SURVEYS»: Зависит от «USER» через внешний ключ «ID\_USER» и от «MEDICAL\_ORGANIZATION» через внешний ключ «ID\_MEDICAL\_ORGANIZATION»
* Сущность «TRAUMA»: Зависит от «USER» через внешний ключ «ID\_USER»

Независимая таблица - это таблица, которая не ссылается на первичный ключ другой таблицы. Независимые таблицы не зависят от других таблиц для своего существования и могут существовать самостоятельно. В независимой таблице могут быть другие таблицы, которые ссылаются на ее первичный ключ, но это не влияет на независимость самой таблицы. Независимыми сущностями являются:

* Сущность «GROUP»
* Сущность «MEDICAL\_ORGANIZATION»
* Сущность «USER\_ROLE»

Выставил в каждой таблице первичные ключи:

* Сущность «GROUP»: Первичный ключ - «ID\_GROUP».
* Сущность «MEDICAL\_ORGANIZATION»: Первичный ключ - «ID\_MEDICAL\_ORGANIZATION».
* Сущность «USER\_ROLE»: Первичный ключ - «ID\_USER\_ROLE».
* Сущность «USER»: Первичный ключ - «ID\_USER».
* Сущность «CHECK\_VACCINE»: Первичный ключ - «ID\_CHECK\_VACCINE».
* Сущность «DATA\_MEDICAL\_CHECKUPS»: Первичный ключ - «ID\_DATA\_MEDICAL\_CHECKUPS».
* Сущность «HOSPITAL\_TREATMENT»: Первичный ключ - «ID\_HOSPITAL\_TREATMENT».
* Сущность «LOGIN\_PASSWORD»: Первичный ключ - «ID\_USER».
* Сущность «OPERATIONS»: Первичный ключ - «ID\_OPERATIONS».
* Сущность «PASS\_CONROLLER»: Первичный ключ - «ID\_PASS\_CONROLLER».
* Сущность «PASSES\_FOR\_ILLNESS»: Первичный ключ - «ID\_PASSES\_FOR\_ILLNESS».
* Сущность «PAST\_INFECTIOUS\_DIASES»: Первичный ключ - «ID\_PAST\_INFECTIOUS\_DIASES».
* Сущность «PREVENTIVE\_VACCINE»: Первичный ключ - «ID\_PREVENTIVE\_VACCINE».
* Сущность «SANITARY\_SPA\_TREATMENT»: Первичный ключ - «ID\_SANITARY\_SPA\_TREATMENT».
* Сущность «SURVEYS»: Первичный ключ - «ID\_SURVEYS».
* Сущность «TRAUMA»: Первичный ключ - «ID\_TRAUMA».

Так же определил внешние ключи таблиц:

* «USER»: Внешние ключи - «ID\_GROUP» (связан с «ID\_GROUP» в таблице «GROUP») и «ID\_USER\_ROLE» (связан с «ID\_USER\_ROLE» в таблице «USER\_ROLE»).
* «CHECK\_VACCINE»: Внешний ключ - «ID\_USER» (связан с «ID\_USER» в таблице «USER»).
* «DATA\_MEDICAL\_CHECKUPS»: Внешние ключи - «ID\_USER» (связан с «ID\_USER» в таблице «USER») и «ID\_MEDICAL\_ORGANIZATION» (связан с «ID\_MEDICAL\_ORGANIZATION» в таблице «MEDICAL\_ORGANIZATION»).
* «HOSPITAL\_TREATMENT»: Внешний ключ - «ID\_USER» (связан с «ID\_USER» в таблице «USER»).
* «LOGIN\_PASSWORD»: Внешний ключ - «ID\_USER» (связан с «ID\_USER» в таблице «USER»).
* «OPERATIONS»: Внешний ключ - «ID\_USER» (связан с «ID\_USER» в таблице «USER»).
* «PASS\_CONROLLER»: Внешний ключ - «ID\_USER» (связан с «ID\_USER» в таблице «USER»).
* «PASSES\_FOR\_ILLNESS»: Внешний ключ - «ID\_USER» (связан с «ID\_USER» в таблице «USER»).
* «PAST\_INFECTIOUS\_DIASES»: Внешний ключ - «ID\_USER» (связан с «ID\_USER» в таблице «USER»).
* «PREVENTIVE\_VACCINE»: Внешние ключи - «ID\_USER» (связан с «ID\_USER» в таблице «USER») и «ID\_MEDICAL\_ORGANIZATION» (связан с «ID\_MEDICAL\_ORGANIZATION» в таблице «MEDICAL\_ORGANIZATION»).
* «SANITARY\_SPA\_TREATMENT»: Внешний ключ - «ID\_USER» (связан с «ID\_USER» в таблице «USER»).
* «SURVEYS»: Внешние ключи - «ID\_USER» (связан с «ID\_USER» в таблице «USER») и «ID\_MEDICAL\_ORGANIZATION» (связан с «ID\_MEDICAL\_ORGANIZATION» в таблице «MEDICAL\_ORGANIZATION»).
* «TRAUMA»: Внешний ключ - «ID\_USER» (связан с «ID\_USER» в таблице «USER»).

Всякое отношение имеет некоторую интерпретацию, передающую смысл представленных в нем данных. Эта интерпретация может быть сформулирована в виде предиката. Предикаты отношений в неформальной записи моей предметной области:

* CHECK\_VACCINE: Пользователь с определенным идентификатором (ID\_USER) проверяется на вакцинацию (ID\_CHECK\_VACCINE) в определенную дату (Date), с температурой тела (BodyTemp), результатом осмотра (ResultInspecion), привит вакциной (Vaccine), имеет разрешение (Permission), и имеет состояние через тридцать минут (StateAfterThirtyMin)."
* DATA\_MEDICAL\_CHECKUPS: пользователь с определенным идентификатором (ID\_USER) проходит медицинский осмотр (ID\_DATA\_MEDICAL\_CHECKUPS) в определенном году (Year) в медицинской организации (ID\_MEDICAL\_ORGANIZATION), при этом осмотр содержит информацию о виде обследования (Survey) и его результате (SurveyResult).
* GROUP: группа с определенным идентификатором (ID\_GROUP) имеет название (GroupName) и количество членов (NumberPeople).
* HOSPITAL\_TREATMENT: пользователь с определенным идентификатором (ID\_USER) проходит лечение в больнице (ID\_HOSPITAL\_TREATMENT) с описанием (Description).
* LOGIN\_PASSWORD: пользователь с определенным идентификатором (ID\_USER) имеет логин (Login) и пароль (Password)."
* MEDICAL\_ORGANIZATION: медицинская организация с определенным идентификатором (ID\_MEDICAL\_ORGANIZATION) имеет название (Name) и адрес (Address).
* OPERATIONS: пользователь с определенным идентификатором (ID\_USER) проходит операцию (ID\_OPERATIONS) с описанием (Operations).
* PASS\_CONROLLER: контролер пропусков с определенным идентификатором (ID\_PASS\_CONROLLER) связан с пользователем (ID\_USER).
* PASSES\_FOR\_ILLNESS: пользователь с определенным идентификатором (ID\_USER) имеет пропуск по болезни (ID\_PASSES\_FOR\_ILLNESS) с определенной датой начала (DateFrom), датой окончания (DateTo) и диагнозом (Diagnosis).
* PAST\_INFECTIOUS\_DIASES: пользователь с определенным идентификатором (ID\_USER) имеет прошедшие инфекционные болезни (ID\_PAST\_INFECTIOUS\_DIASES) с датой (Date) и описанием (Description).
* PREVENTIVE\_VACCINE: пользователь с определенным идентификатором (ID\_USER) получает профилактическую вакцину (ID\_PREVENTIVE\_VACCINE) в определенной медицинской организации (ID\_MEDICAL\_ORGANIZATION), вакцина имеет название (Vaccine) и дату (Date).
* SANITARY\_SPA\_TREATMENT: пользователь с определенным идентификатором (ID\_USER) проходит санаторно-курортное лечение (ID\_SANITARY\_SPA\_TREATMENT) с описанием (Description)."
* SURVEYS: пользователь с определенным идентификатором (ID\_USER) проходит обследование (ID\_SURVEYS) в медицинской организации (ID\_MEDICAL\_ORGANIZATION) в определенном году (Year), обследование содержит информацию о виде обследования (Survey) и его результате (SurveyResult).
* TRAUMA: пользователь с определенным идентификатором (ID\_USER) получает травму (ID\_TRAUMA) с описанием (Description).
* USER: пользователь с определенным идентификатором (ID\_USER) имеет роль (ID\_USER\_ROLE), фамилию (Surname), имя (Name), отчество (Patronymic), дату рождения (DateBirth), домашний адрес (HomeAddress), проживает в общежитии (Hostel), является членом группы (ID\_GROUP), дату зачисления (DateEnrollment), цвет (Color), и название изображения (ImageName).
* USER\_ROLE: роль пользователя с определенным идентификатором (ID\_USER\_ROLE) имеет название (NameRole).

**Составление глоссариев и спецификаций**

Глоссарий – это словарь, в котором содержатся толкования всех слов, которые могут быть незнакомы читателю. Глоссарий значительно облегчает восприятие текста, так как человек в любой момент имеет возможность обратиться к словарю и проверить значение определённого термина.

После выявлений предикатов отношений был сформирован глоссарий сущностей, содержащий наименование сущности и её описание. Глоссарии могут формироваться разными способами: при помощи мастера отчётов в программах для проектирования и документирования баз данных или вручную в текстовых редакторах. Исходя из логической модели мне удалось сформировать глоссарий сущностей и глоссарий атрибутов (см. Приложение 1).

После составления глоссария сущностей был сформирован глоссарий атрибутов, содержащий наименование атрибута, наименование сущности атрибута, тип и описание атрибута (см. Приложение 2).

**Создание физической модели базы данных**

Физическая модель базы данных представляет собой модель данных, которая описывает, как данные будут храниться в базе данных, и содержит все необходимые сведения для создания базы данных в конкретной системе управления базами данных (СУБД). Эта модель включает в себя описание таблиц, столбцов, индексов, первичных и внешних ключей, типов данных, ограничений и других элементов, необходимых для создания базы данных. Физическая модель базы данных является конечным результатом процесса проектирования базы данных и используется для создания скриптов SQL, необходимых для создания базы данных.

База данных является частью системы базы данных, которая также включает в себя программное обеспечение общего назначения, известное как система управления базой данных (СУБД), используемое для управления базой данных, соответствующее оборудование и персонал. Все эти компоненты работают вместе, чтобы обеспечить эффективное хранение, управление и извлечение данных из базы данных.

Каждая СУБД должна удовлетворять следующим требованиям:

* Надежность. СУБД должна обеспечивать надежное хранение данных и предотвращать их потерю или повреждение.
* Безопасность. СУБД должна обеспечивать безопасность данных, предоставляя механизмы аутентификации и авторизации, а также шифрование и контроль доступа.
* Масштабируемость. СУБД должна быть способна масштабироваться в соответствии с растущими потребностями в хранении и обработке данных.
* Производительность. СУБД должна обеспечивать высокую производительность при обработке запросов и транзакций.
* Согласованность. СУБД должна обеспечивать согласованность данных, предотвращая конфликты и несоответствия при одновременном доступе нескольких пользователей.
* Управление данными. СУБД должна предоставлять средства для управления данными, включая создание, изменение, удаление и восстановление данных.
* Поддержка стандартов. СУБД должна поддерживать стандарты SQL и другие стандарты баз данных, чтобы обеспечить совместимость с другими системами и приложениями.
* Удобство использования. СУБД должна быть легкой в использовании и обеспечивать интуитивно понятный интерфейс для администрирования и разработки приложений.

Система с базой данных состоит из следующих компонентов:

* База данных. это набор связанных таблиц, хранящих данные.
* Система управления базой данных (СУБД). это программное обеспечение, которое управляет базой данных и предоставляет интерфейс для взаимодействия с ней.
* Приложения. это программы, которые используют базу данных для хранения и извлечения данных.
* Пользователи. это люди, которые взаимодействуют с базой данных через приложения.
* Администраторы баз данных. это люди, которые отвечают за управление базой данных, включая ее проектирование, развертывание, настройку, мониторинг и обслуживание.
* Оборудование. это серверы, хранилища данных, сети и другие физические компоненты, необходимые для работы системы с базой данных.
* Программное обеспечение поддержки. это дополнительные программные продукты, такие как средства резервного копирования и восстановления, средства мониторинга производительности и средства безопасности.

Система-хост предоставляет доступ к строкам данных. Система управления базами данных (СУБД) генерирует запросы, используя функциональные возможности системы управления файлами системы-хоста, для обслуживания различных приложений. СУБД является дополнительным уровнем программного обеспечения, который работает над программным обеспечением системы-хоста.

При сравнении возможностей, достоинств и недостатков разных систем управления базами данных мой выбор пал на Microsoft SQL Server. Выбор Microsoft SQL Server обусловлен рядом стратегических и технических факторов, которые существенно влияют на обработку и хранение медицинских данных в контексте медицинского кабинета колледжа.

1. Стандартизация и Интеграция. Использование SQL Server соответствует стандартам, что упрощает интеграцию с другими продуктами Microsoft, обеспечивая единый стандарт для разработки и управления данными в рамках цифровой экосистемы Microsoft.

2. Безопасность и Соблюдение. SQL Server предоставляет мощные механизмы безопасности и соответствует стандартам соблюдения законодательства в области медицинской конфиденциальности, что является приоритетом при обработке медицинских данных.

3. Производительность и Масштабируемость. Высокая производительность SQL Server, особенно в контексте транзакционных операций и запросов, обеспечивает эффективную обработку и доступ к данным, что крайне важно для оперативной работы медицинского кабинета.

4. Гибкость и Надежность. Система предоставляет гибкие инструменты для разработки и масштабируемость, позволяя легко адаптироваться к изменяющимся требованиям и обеспечивая надежность в хранении ценных медицинских данных.

5. Поддержка и Обучение. Большое сообщество пользователей SQL Server, обширная документация и наличие профессиональных ресурсов создают благоприятные условия для обучения и развития навыков в работе с данной СУБД.

Общий выбор SQL Server вытекает из стремления обеспечить безопасность, эффективность и совместимость с современными технологиями при обработке и управлении медицинской информацией в медицинском кабинете колледжа.

После выполнения соединения системы управления базами данных с сервером была создана физическая модель автоматизированной информационной системы «Контроль состояния здоровья студентов ГАПОУ СО «Камышловский педагогический колледж».

После создания физической модели базы данных удалось выявить следующие схемы реляционных таблиц:

* + - * USER (ID\_USER, ID\_USER\_ROLE, Surname, Name, Patronymic, DateBirth, HomeAddress, Hostel, ID\_GROUP, DateEnrollment, Color, ImageName);
      * GROUP (ID\_GROUP, GroupName, NumberPeople);
      * CHECK\_VACCINE (ID\_CHECK\_VACCINE, ID\_USER, Date, BodyTemp, ResultInspecion, Vaccine, Permission, StateAfterThirtyMin);
      * DATA\_MEDICAL\_CHECKUPS (ID\_DATA\_MEDICAL\_CHECKUPS, ID\_USER, ID\_MEDICAL\_ORGANIZATION, Year, Survey, SurveyResult);
      * HOSPITAL\_TREATMENT (ID\_HOSPITAL\_TREATMENT, ID\_USER, Description);
      * LOGIN\_PASSWORD (Login, Password, ID\_USER);
      * MEDICAL\_ORGANIZATION (ID\_MEDICAL\_ORGANIZATION, Name, Address);
      * OPERATIONS (ID\_OPERATIONS, ID\_USER, Operations);
      * PASS\_CONROLLER (ID\_PASS\_CONROLLER, ID\_USER);
      * PASSES\_FOR\_ILLNESS (ID\_PASSES\_FOR\_ILLNESS, ID\_USER, DateFrom, DateTo, Diagnosis);
      * PAST\_INFECTIOUS\_DIASES (ID\_PAST\_INFECTIOUS\_DIASES, ID\_USER, Date, Description);
      * PREVENTIVE\_VACCINE (ID\_PREVENTIVE\_VACCINE, ID\_USER, ID\_MEDICAL\_ORGANIZATION, Vaccine, Date);
      * SANITARY\_SPA\_TREATMENT (ID\_SANITARY\_SPA\_TREATMENT, ID\_USER, Description);
      * SURVEYS (ID\_SURVEYS, ID\_USER, Year, Survey, SurveyResult, ID\_MEDICAL\_ORGANIZATION);
      * TRAUMA (ID\_TRAUMA, ID\_USER, Description);
* USER\_ROLE (ID\_USER\_ROLE, NameRole);

На следующем этапе проектирования физической модели базы данных исходные файлы готовятся для импорта. Этот процесс может включать форматирование неструктурированных файлов для использования их в импорте в базу данных. Были созданы файлы для импорта данных с индивидуальными проектами, курсовыми работами/проектами и выпускными квалификационными работами.

После того, как исходные файлы были подготовлены для импорта, выполняется процесс импорта данных в базу данных. Для этого создается временная база данных, в которую импортируются данные для последующего переноса в основную базу данных. С помощью мастера импорта и экспорта SQL Server производится импорт файлов формата \*.csv, \*.xls и \*.txt в временную базу данных. Затем, используя язык структурирования запросов SQL, выполняется копирование данных в основную базу данных.

После того, как была выбрана система управления базами данных, выявлены схемы реляционных таблиц, реализованы функции поддержания внутренних ограничений целостности и выполнен импорт данных, было принято решение создать скрипт для переноса и внедрения базы данных на другой SQL Server.

**Проектирование приложений конечных пользователей**

**Выбор среды разработки приложений для ИС**

Из множества существующих языков программирования я выбрал C#, поскольку он является одним из самых популярных и широко используемых языков программирования во всем мире. Этот язык предоставляет богатую библиотеку классов и инструментов для разработки приложений, поддерживает объектно-ориентированное программирование и имеет хорошую интеграцию с системами управления базами данных. Кроме того, он широко используется в разработке программного обеспечения для Windows и имеет большое сообщество разработчиков, что обеспечивает наличие многочисленных ресурсов и инструментов для обучения и поддержки. Все это делает C# идеальным выбором для моего проекта. Чтобы программировать на C# необходимо использовать интегрированную среду разработки, которая должна быть совместима с операционной системой. Выбор подходящей среды разработки является важным и актуальным вопросом, так как он влияет на эффективность и удобство работы программиста.

Интегрированная среда разработки (Integrated Development Environment, IDE) представляет собой комплексное программное обеспечение, которое объединяет в себе необходимые инструменты для разработки программного обеспечения. Она включает в себя текстовый редактор, компилятор или интерпретатор, средства автоматизации сборки и отладчик. Кроме того, интегрированная среда разработки может содержать инструменты для интеграции с системами управления версиями и другие полезные утилиты. Некоторые интегрированные среды разработки предназначены для работы только с одним языком программирования, в то время как большинство современных IDE поддерживают несколько языков. Выбор подходящей интегрированной среды разработки является важным шагом для разработчика, так как она значительно облегчает процесс разработки и улучшает его эффективность.

Для выбора наиболее подходящей среды разработки для программирования на языке C# было проведено сравнение нескольких популярных интегрированных сред разработки, таких как Microsoft Visual Studio, JetBrains Rider, Visual Studio Code и Sharpdevelop. Были рассмотрены преимущества и недостатки каждой из них, чтобы сделать обоснованное решение о выборе наиболее подходящей среды разработки.

Microsoft Visual Studio является одной из самых популярных и широко используемых интегрированных сред разработки для программирования на языке C#. Она предоставляет широкий спектр возможностей для разработки, отладки и тестирования приложений, а также имеет удобный и настраиваемый интерфейс. Кроме того, Visual Studio поддерживает работу с различными системами управления версиями и предоставляет множество полезных инструментов для разработки.

JetBrains Rider является относительно новым продуктом компании JetBrains, который также предназначен для программирования на языке C#. Эта интегрированная среда разработки имеет множество преимуществ, таких как высокая производительность, настраиваемый интерфейс, поддержка работы с различными системами управления версиями и множество полезных инструментов для разработки. Кроме того, JetBrains Rider имеет встроенный профайлер и инструменты для анализа кода, которые помогают оптимизировать производительность приложения.

Visual Studio Code является легковесной и гибкой интегрированной средой разработки, которая также поддерживает программирование на языке C#. Она предоставляет множество полезных расширений для разработки, отладки и тестирования приложений, а также имеет удобный и настраиваемый интерфейс. Visual Studio Code также поддерживает работу с различными системами управления версиями и предоставляет множество полезных инструментов для разработки.

Sharpdevelop является бесплатной и открытой интегрированной средой разработки для программирования на языке C#. Она предоставляет множество полезных инструментов для разработки, отладки и тестирования приложений, а также имеет удобный и настраиваемый интерфейс. Sharpdevelop также поддерживает работу с различными системами управления версиями и предоставляет множество полезных инструментов для разработки.

В итоге, после сравнения преимуществ и недостатков каждой из интегрированных сред разработки, было принято решение выбрать Microsoft Visual Studio в качестве основной среды разработки для программирования на языке C#. Это решение было обосновано широким спектром возможностей, предоставляемых Visual Studio, ее высокой производительностью и надежностью, а также наличием множества полезных инструментов для разработки, отладки и тестирования приложений.

**Проектирование с использованием UML диаграмм**

UML (Unified Modeling Language) - это стандартный язык моделирования программного обеспечения, используемый для визуального представления структуры и поведения программных систем. UML включает в себя набор графических диаграмм, которые используются для моделирования различных аспектов программного обеспечения, таких как классы, объекты, взаимодействия, состояния и процессы.

UML позволяет разработчикам создавать визуальные модели программного обеспечения, которые помогают им лучше понимать архитектуру системы и ее взаимосвязи. Это также облегчает обмен информацией между разработчиками, позволяя им легко обмениваться моделями и диаграммами.

Диаграмма деятельности.

Диаграмма деятельности UML обеспечивает визуальное представление бизнес-процессов, поддерживая моделирование последовательностей, условий, параллельных операций, циклов и потоков данных. Она улучшает понимание бизнес-логики, решений и взаимодействия между компонентами системы, делая её полезной для анализа и документирования бизнес-процессов.

Диаграмма деятельности UML представляет взаимодействие между Информационной системой и Базой данных при создании медицинского осмотра для студента. Информационная система начинает с запроса на создание медицинского осмотра и поиска студента в базе данных. Если студент не найден, система сообщает об этом и возвращает к запросу на поиск. В случае наличия студента, система создает форму для заполнения медицинского осмотра, после чего происходит заполнение данных и запрос на сохранение записи в базе данных. База данных проверяет соответствие данных формату и ограничениям. Если данные не соответствуют, система сообщает об ошибке и возвращает к заполнению формы. В случае успешного сохранения, система завершает процесс, сообщая об успешном сохранении данных.

Диаграмма последовательности.

Диаграмма последовательности UML предоставляет визуальное представление последовательности взаимодействия между объектами в системе. Она позволяет визуализировать и описывать шаги выполнения сценария, отслеживать временные аспекты, выделять параллельные потоки выполнения, моделировать альтернативные пути и анализировать использование системы, что делает её полезной для документирования процессов и обнаружения потенциальных проблем в системе.

Диаграмма последовательности UML описывает взаимодействие между тремя сущностями: Пользователь, Информационная система и База данных в контексте запроса на создание медицинского осмотра.

Описание последовательности событий:

* Начало ЖЦ пользователя (User's Lifecycle Start): пользователь инициирует Жизненный Цикл (ЖЦ), представляя запрос на создание медицинского осмотра.
* Начало ЖЦ информационной системы (System's Lifecycle Start): информационная система получает запрос от пользователя о создании медицинского осмотра.
* Начало ЖЦ базы данных (Database's Lifecycle Start): информационная система инициирует запрос на получение данных о студенте в базе данных.
* База данных возвращает данные о студенте (Database Returns Student Data): база данных возвращает необходимые данные о студенте.
* Конец ЖЦ базы данных (Database's Lifecycle End): база данных завершает свой ЖЦ после возвращения данных.
* Информационная система отображает данные о студенте (System Displays Student Data): информационная система принимает данные от базы данных и отображает их пользователю.
* Возврат в ЖЦ пользователя (Return to User's Lifecycle): пользователь приступает к заполнению данных о медицинском осмотре.
* Возврат в ЖЦ информационной системы (Return to System's Lifecycle): после заполнения данных пользователь инициирует запрос на сохранение медицинского осмотра.
* Начало ЖЦ Базы данных (Database's Lifecycle Start): информационная система направляет запрос на сохранение медицинского осмотра в базу данных.
* База данных возвращает результаты сохранения (Database Returns Save Results):
* База данных возвращает результаты сохранения медицинского осмотра.
* Конец ЖЦ базы данных (Database's Lifecycle End):
* База данных завершает свой ЖЦ после сохранения данных.
* Возврат в ЖЦ информационной системы (Return to System's Lifecycle):
* Информационная система получает результаты сохранения и сообщает об этом пользовате-лю.
* Система сообщает о результате сохранения (System Reports Save Results):
* Информационная система сообщает пользователю о результатах сохранения медицинского осмотра.
* Возврат в ЖЦ пользователя (Return to User's Lifecycle):
* Пользователь завершает ЖЦ после получения информации о результате сохранения.

Эта диаграмма последовательности иллюстрирует взаимодействие между пользователями, информационной системой и базой данных в процессе создания медицинского осмотра.

**Проектирование пользовательского интерфейса**

**Программная реализация информационной системы**

Архитектура программного продукта — это структурное описание и организация программного обеспечения, которое определяет его компоненты, их взаимосвязи и взаимодействие с окружающей средой. Это набор ключевых принципов и решений, направленных на обеспечение эффективности, устойчивости, масштабируемости и других характеристик программной системы.

Архитектура программного продукта "Контроль состояния здоровья студентов ГАПОУ СО «Камышловский педагогический колледж»":

* Клиентская часть. Интерфейс для сотрудников медицинского кабинета и студентов разработан с использованием технологии WPF (Windows Presentation Foundation) от Microsoft. WPF обеспечивает создание современных, гибких и интерактивных пользовательских интерфейсов для Windows-приложений.
* Серверная часть. База данных SQL Server является центральным элементом серверной части. Здесь хранятся электронные медицинские карточки студентов и другие связанные с ними данные. Серверная часть обеспечивает обработку запросов от клиентов, обеспечивает безопасность данных и осуществляет управление базой данных.
* Бизнес-логика. Программный продукт реализует бизнес-логику, которая включает в себя обработку данных, проведение медицинских осмотров, валидацию вводимых данных и формирование отчетов. Бизнес-логика выполняется на сервере для обеспечения единообразия и безопасности обработки данных.
* Сетевая архитектура. Программный продукт использует сетевую архитектуру для обеспечения взаимодействия между клиентской и серверной частями. Клиенты имеют доступ к серверу через защищенное соединение, обеспечивающее конфиденциальность и целостность передаваемых данных.
* Безопасность. Реализованы меры безопасности, такие как авторизация и аутентификация пользователей, шифрование данных при передаче и хранении. Также предусмотрены ролевые права доступа к данным, обеспечивающие контролируемый доступ к конфиденциальной медицинской информации.
* Масштабируемость. Архитектура продукта обеспечивает масштабируемость, что позволяет эффективно управлять ростом данных и нагрузкой при необходимости расширения функциональности или увеличения числа пользователей.

Архитектура программного продукта с использованием WPF в клиентской части создана с учетом требований к безопасности, производительности и удобства использования, обеспечивая эффективное взаимодействие медицинского персонала и студентов с информационной системой.

**Разработка интерфейса приложений**

**Структура интерфейса**

Главное меню. Меню, расположенное в левой части интерфейса, содержит основные категории и функции системы. Здесь пользователь может найти доступ к основным разделам и действиям (см. Рисунок 5).

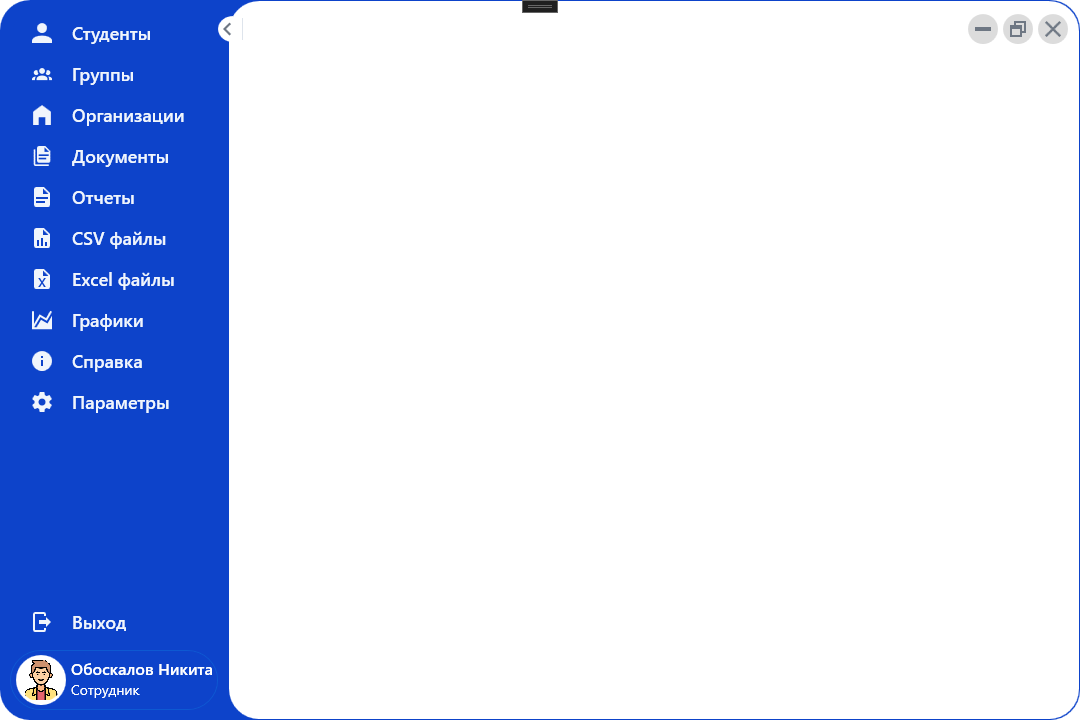


Рисунок 5 - Меню приложения

Область рабочего пространства. Основное окно, где отображается основная информация и контент. В этой области пользователь взаимодействует с данными, выполняет задачи и просматривает результаты операций (см. Рисунок 6).

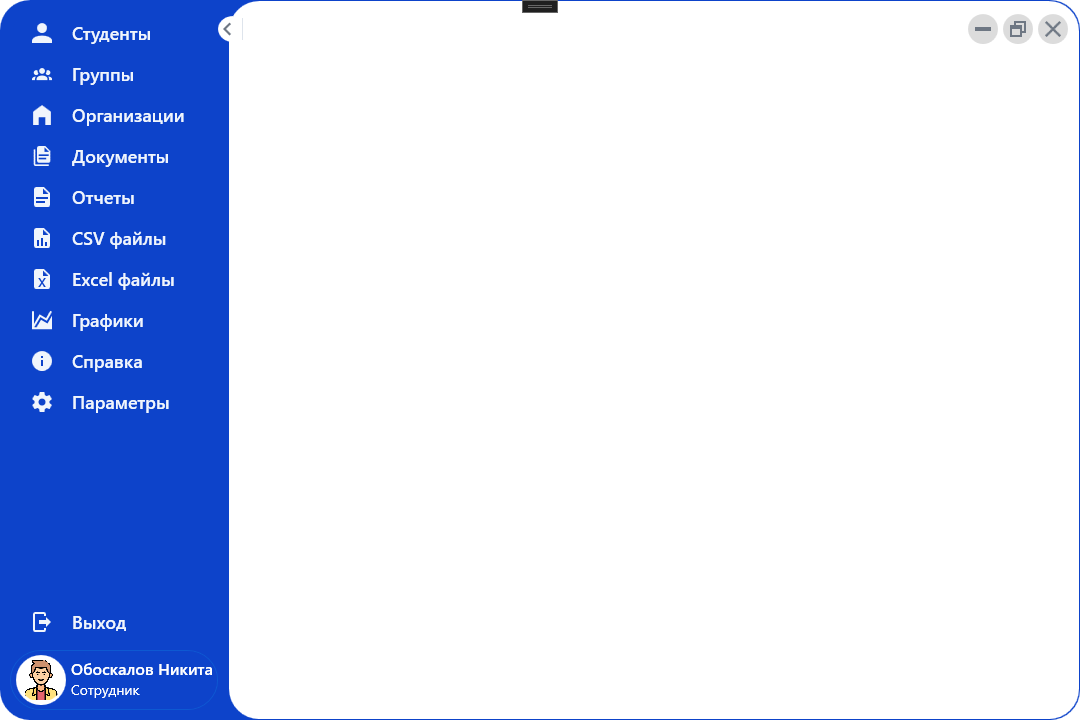


Рисунок 6 - Панель навигации

Формы и диалоговые окна. Элементы, предоставляющие пользователю возможность ввода данных или выбора опций. Формы могут использоваться для создания, редактирования или поиска информации (см. Рисунок 7).

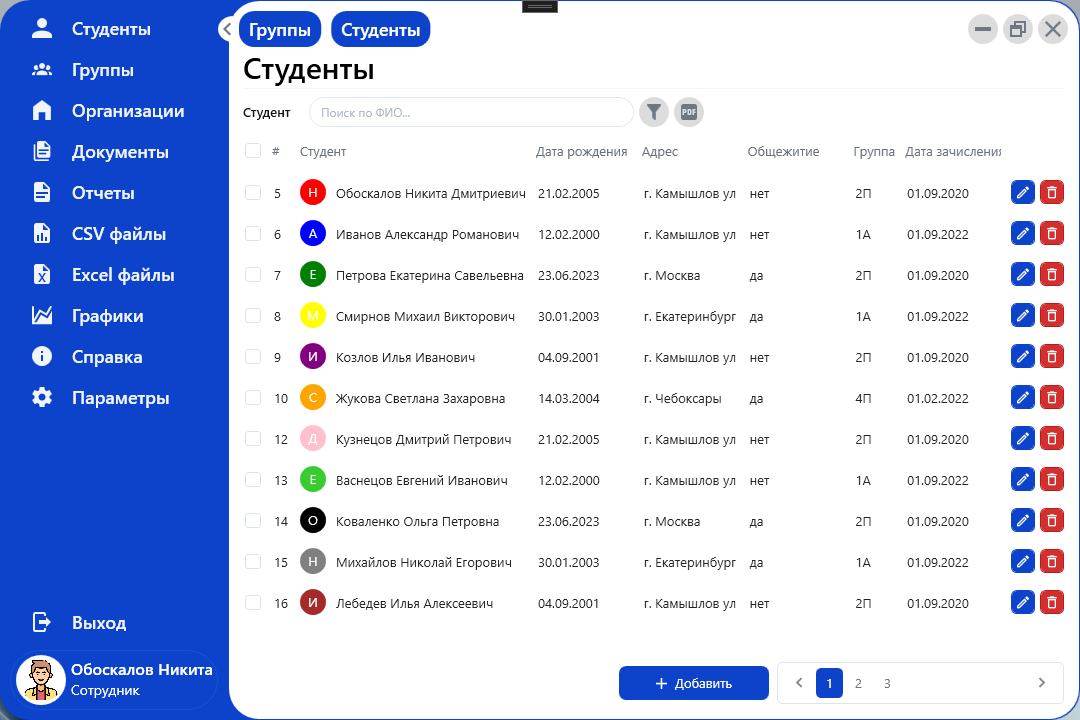


Рисунок 7 - Формы и диалоговые окна

Элементы управления. Кнопки, чекбоксы, радиокнопки, выпадающие списки и другие элементы, предоставляющие пользователю средства взаимодействия с системой (см. Рисунок 8).

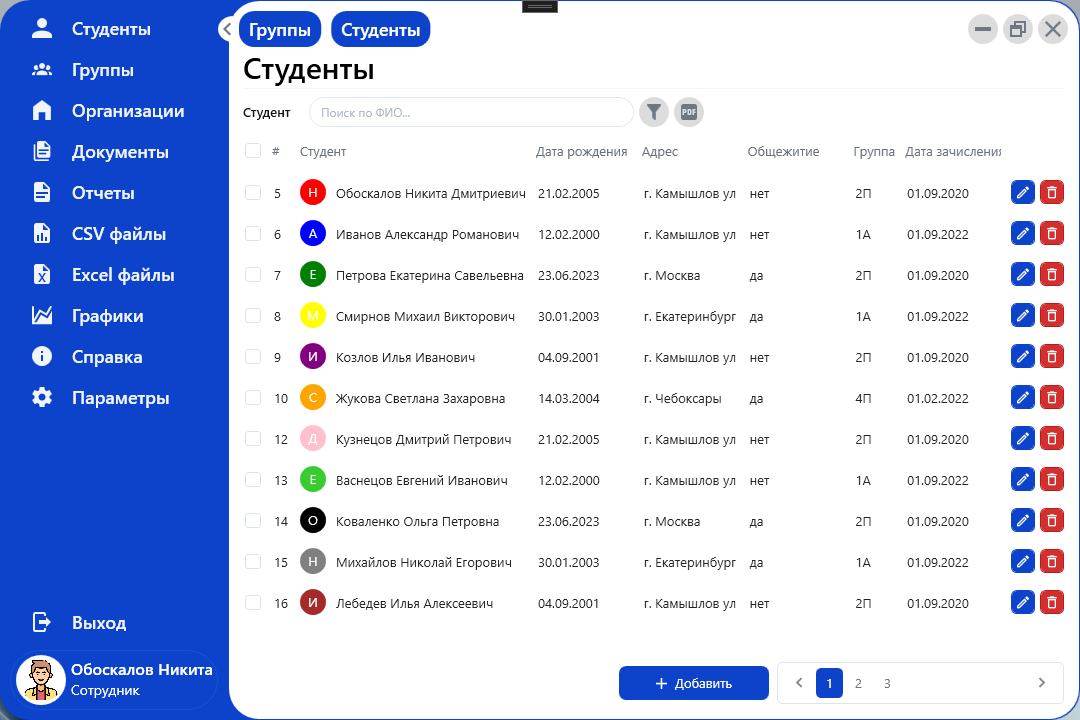


Рисунок 8 - Элементы управления

**Описание меню**

Меню в программном интерфейсе представляет собой структурированный набор команд, опции и функций, предоставляемых пользователю для управления программой или системой. Меню организовано в виде иерархии, где элементы разделены по категориям, что облегчает пользователям навигацию и нахождение необходимых функций. Основные компоненты и характеристики меню:

* Студенты. Элемент меню открывает страницу студентов, на которой выводится полный список студентов из базы данных с возможностью постраничного просмотра. Пользователь может легко перелистывать страницы, а также использовать кнопки для удаления, редактирования и добавления студента. Элементы фильтрации позволяют настраивать отображение данных по заданным критериям.
* Группы. Элемент меню открывает страницу групп, на которой пользователю предоставляется список всех групп и студентов, входящих в выбранную группу из базы данных. Кнопки редактирования, добавления и удаления групп упрощают администрирование данных. Элементы фильтрации предоставляют возможность настройки отображения данных по выбранным параметрам.
* Организации. На странице медицинских организаций представлены все организации из базы данных. Пользователь может легко управлять данными с помощью кнопок удаления, редактирования и добавления организаций. Элементы фильтрации обеспечивают персонализированный просмотр данных.
* Документы. Страница документов предоставляет пользователю удобный доступ ко всем документам, доступным для него. Информация о документах представлена в удобной форме.
* Отчеты. Страница отчетов предоставляет пользователю перечень доступных отчетов. Пользователь может просматривать и загружать отчеты, необходимые ему для работы.
* CSV файлы. На странице выбора CSV файлов пользователь может загружать CSV файлы с компьютера для вывода данных. Это предоставляет удобный способ импорта информации из внешних источников.
* Excel файлы. Страница выбора Excel файлов позволяет пользователю выбирать и загружать Excel файлы с компьютера для дополнительного анализа данных.
* Графики. Элемент меню "Графики" дает возможность просмотра динамического списка графиков, обеспечивая визуализацию данных.
* Справка. Страница со списком файлов, полезных для пользователя, предоставляет доступ к справочной информации, руководствам и другим документам, помогающим в использовании приложения.
* Параметры. Страница настроек приложения предоставляет пользователю возможность выбора и изменения параметров приложения, что позволяет настраивать интерфейс и функциональность под свои предпочтения.

Выход. Элемент меню "Выход" открывает окно авторизации пользователя, позволяя ему безопасно завершить сеанс работы в приложении.

**Реализация функций поиска и фильтрации данных**

Модуль поиска и фильтрации представляет собой метод ApplyFilters, который отвечает за применение фильтров к данным и обновление отображаемой информации в DataGrid. Рассмотрим его основные компоненты:

1. Очистка DataGrid:

StudentsDataGrid.ItemsSource = null; - Очищает текущее содержимое DataGrid перед применением новых фильтров.

1. Формирование условий фильтрации:

Создаются списки conditions и parameters, в которых будут храниться условия фильтрации и их параметры для последующего использования в SQL-запросе.

1. Фильтрация по возрасту, общежитию, группе и текстовому поиску:

Используются значения, выбранные пользователем в ComboBox'ах и TextBox'е, для формирования соответствующих условий фильтрации.

1. Составление SQL-запроса с учетом всех условий:

Создается SQL-запрос, учитывающий все условия фильтрации, включая возраст, общежитие, группу и текстовый поиск.

1. Выполнение SQL-запроса и получение данных:

Вызывается метод ExecuteSqlParameters для выполнения SQL-запроса с использованием параметров.

1. Заполнение коллекции студентов данными из запроса:

Происходит обход полученных данных и заполнение коллекции allStudents объектами класса Student на основе результатов SQL-запроса.

1. Обновление общего количества записей и количество страниц:

Обновляются переменные totalRecords и totalPages в зависимости от количества записей и количества записей на странице.

1. Обновление DataGrid и кнопок страниц:

Вызываются методы FillStudentsDataGrid() и UpdatePageButtons() для обновления отображаемых данных в DataGrid и кнопок управления страницами.

1. Информирование пользователя о результатах:

Используются диалоговые окна MessageBox для информирования пользователя о результатах операции, например, успешном удалении студента или возможной ошибке.

1. Обработка исключений:

Реализован блок обработки исключений для отлавливания возможных ошибок при выполнении SQL-запросов и обновлении данных.

Этот метод представляет собой важную часть логики работы с данными и позволяет пользователям эффективно фильтровать и искать информацию о студентах в базе данных.

**Реализация функций просмотра и генерации отчётов**

Хотя в настоящее время существует множество готовых решений для генерации отчетов, которые могут быть интегрированы в информационные системы, разработчикам программного обеспечения иногда не хватает их функциональности для реализации поставленных задач. Кроме того, такое готовое решение может не соответствовать технологическому стеку разрабатываемого проекта. В этих случаях разработчики вынуждены создавать собственную подсистему генерации отчетов для своей информационной системы.

Зная структуру отчёта, было решено приступить к созданию набора данных, который будет выступать источником данных для отчёта. Используя элементы и данные из набора данных удалось реализовать шаблоны отчётов, формирующиеся программно в АИС. В итоге были разработаны отчёты по всем таблицам.

Для того, чтобы сгенерировать отчёт по определённому параметру, необходимо выбрать из комбинированного списка группу и нажать на кнопку «Экспорт в Word» или «Экспорт в Excel».

**Реализация дополнительных функций приложений**

В автоматизированной информационной системе предусмотрены дополнительные функции, облегчающие работу с данными. Одной из таких функций является постраничный просмотр списка. Эта функция позволяет выводить информацию для пользователя в определенном количестве, например, по тридцать записей на странице. Для переключения между страницами предусмотрены специальные кнопки в интерфейсе. Рядом с кнопками отображается максимальное количество страниц и номер текущей страницы, на которой находится пользователь. Таким образом, пользователь может легко перемещаться по списку и находить нужную информацию, не перегружая себя избыточным количеством данных. Кроме того, постраничный просмотр позволяет оптимизировать работу системы и уменьшить время загрузки данных. (см. Рисунок 9).

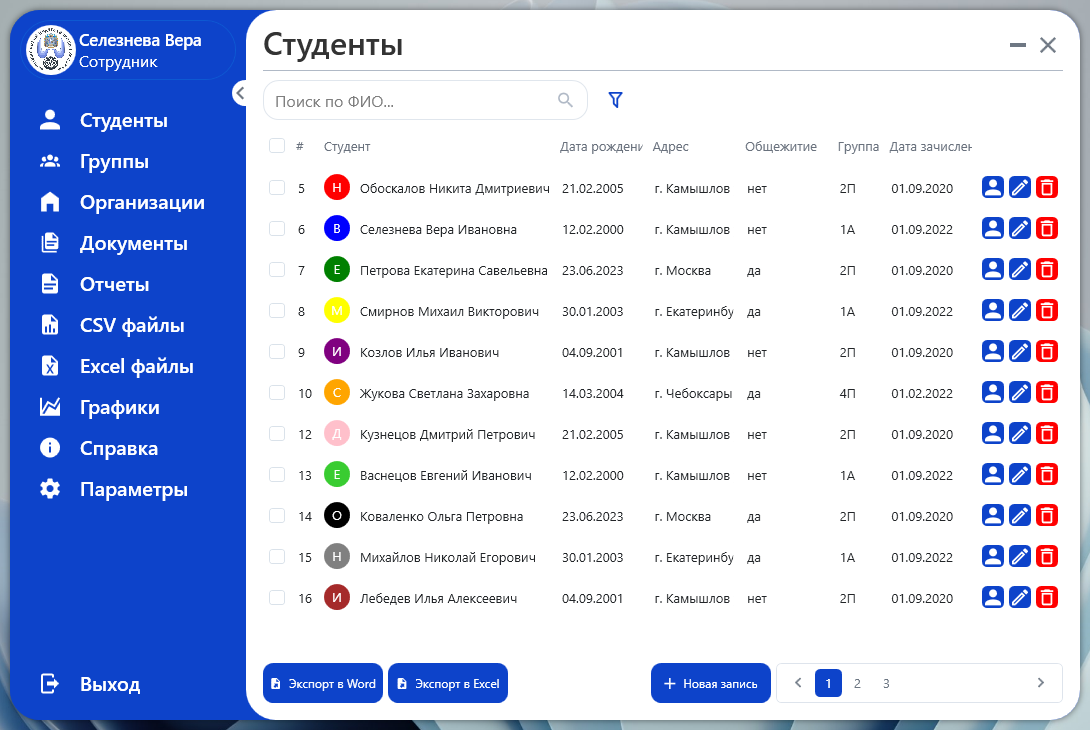


Рисунок 9 - Использование постраничного просмотра

Следующая дополнительная функция АИС: представление данных в виде графиков. Такой способ даёт возможность рассмотреть данные графически. Удалось реализовать график, содержащий статистику об количестве человек в группе (см. Рисунок 10).

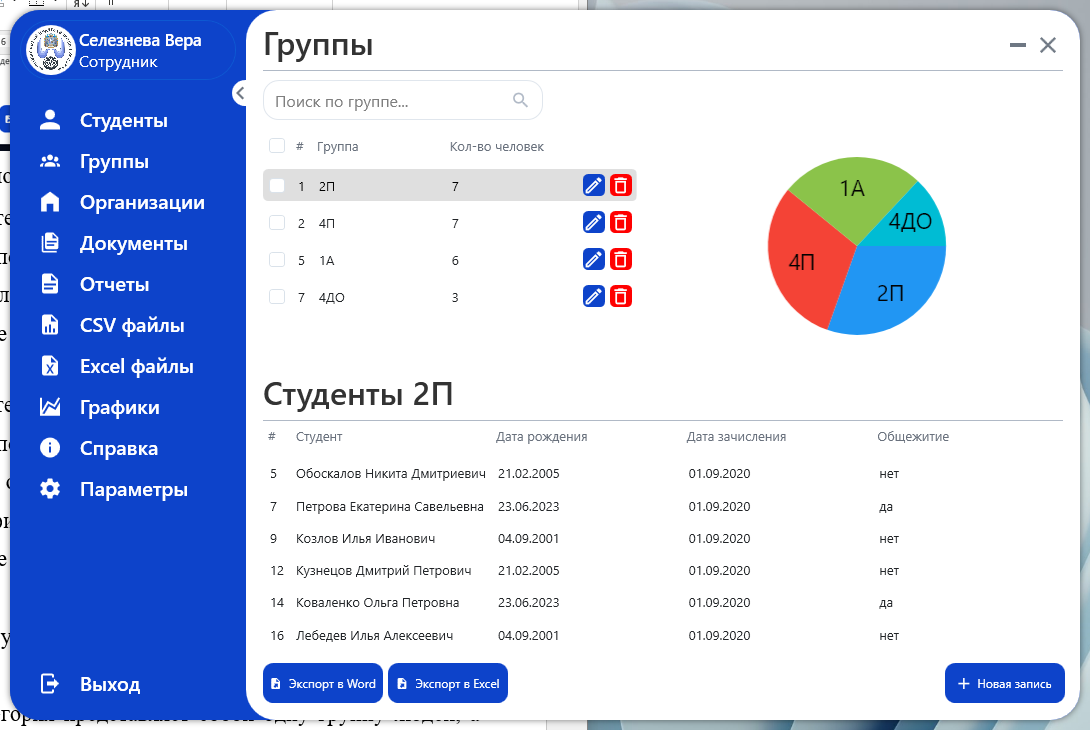


Рисунок 10 - График

График в виде круга представляет собой диаграмму, которая демонстрирует процентное соотношение различных категорий данных. В нашем случае, каждая категория представляет собой одну группу людей, а величина сектора круга соответствует количеству человек в этой группе.

На графике в виде круга, каждая группа людей представлена своим цветом и подписью, которая указывает название группы. Размер сектора круга пропорционален количеству человек в группе. Таким образом, пользователь может легко оценить соотношение количества людей в различных группах.

Использование графиков в виде круга является эффективным способом визуализации данных, так как позволяет легко оценить процентное соотношение различных категорий данных. Кроме того, такой способ представления данных является более наглядным и удобным для восприятия, чем табличные данные.

Таким образом, реализация графика в виде круга в АИС позволяет пользователям более эффективно анализировать данные и принимать обоснованные решения.

В следующей функциональной возможности АИС предусмотрена запись файлов Excel. При создании этой функции было учтено многообразие форматов Excel-документов. Эта функция позволяет пользователям сохранять данные в файлы Excel, не покидая приложение. Запись означает сохранение данных в документ Excel.

После того, как данные были получены из базы данных, пользователь может работать с ними и сохранять в нужном формате файла. При сохранении данных открывается окно, в котором пользователь может выбрать место сохранения файла из списка доступных локаций, а также изменить название сохраняемого документа и выбрать нужный формат файла. Эта функция позволяет пользователю гибко управлять данными и сохранять их в удобном для него формате.

**Тестирование приложений и отладка программных модулей**

Отладка является важным этапом разработки программного обеспечения, который помогает выявлять и исправлять ошибки в коде. Инструменты отладки для приложений на языке C# с использованием технологии WPF и базы данных SQL Server:

Отладчик — это мощный инструмент для поиска и исправления ошибок в коде. Позволяет программистам следить за выполнением программы, шаг за шагом, устанавливать точки останова для анализа состояния приложения в конкретные моменты времени и просматривать значения переменных. Это позволяет выявлять и исправлять ошибки на ранних этапах разработки.

Логирование. Логирование является важным инструментом для отслеживания выполнения программы и выявления ошибок. Путем вывода сообщений в логи программисты могут отслеживать последовательность выполнения программы, а также записывать значения переменных или другую важную информацию. Инструменты анализа логов, такие как ELK Stack (Elasticsearch, Logstash, Kibana), позволяют агрегировать, анализировать и визуализировать данные логов для обнаружения проблем и ошибок.

Профилирование — это процесс измерения производительности приложения с целью выявления узких мест и оптимизации кода. Инструменты типа Profiler позволяют анализировать производительность приложения, выявляя участки кода, где программа тратит больше всего времени или ресурсов. Это позволяет оптимизировать код для повышения производительности и улучшения пользовательского опыта.

Хотя отладка является важной частью разработки программного обеспечения, существуют инструменты и подходы, которые могут помочь снизить потребность в отладке или предотвратить появление ошибок на ранних этапах разработки. Вот несколько таких инструментов:

Статические анализаторы кода (Code Analysis Tools). Используются для автоматической проверки и анализа исходного кода на предмет потенциальных проблем, ошибок и нарушений стандартов кодирования. Помогают выявить проблемы до запуска приложения, что позволяет предотвратить появление многих ошибок.

Модульное тестирование (Unit Testing). Позволяет проверить каждый модуль программы на корректность работы независимо от других модулей. При наличии надлежащего покрытия модульными тестами можно заметить ошибки на ранних этапах разработки, еще до того, как код достигнет стадии отладки.

Code Reviews. Проведение код-ревью помогает выявить потенциальные проблемы и ошибки на ранних этапах разработки. При этом другие члены команды могут предложить свои идеи и улучшения, что также способствует повышению качества кода.

Использование стабильных и проверенных библиотек и фреймворков. Использование уже проверенных и отлаженных библиотек и фреймворков может снизить вероятность возникновения ошибок в приложении.

Принципы SOLID и чистый код (Clean Code). Соблюдение принципов SOLID и написание чистого, понятного кода может существенно снизить вероятность появления ошибок и упростить отладку.

Использование этих инструментов и подходов в процессе разработки помогает снизить зависимость от отладки и обеспечить создание более надежного и качественного программного обеспечения.

Для отладки была выбрана форма авторизации пользователей, поскольку она включает в себя процесс аутентификации и авторизации, который часто требует работы с конфиденциальными данными пользователей. Более того, в этом процессе используется класс шифрования для защиты чувствительных данных, таких как пароли.

Отладка этой формы позволяет проверить правильность реализации класса шифрования и его взаимодействие с другими компонентами приложения. Это важно, потому что неправильная работа шифрования может привести к утечке конфиденциальной информации или уязвимостям безопасности.

При отладке формы авторизации можно также проверить правильность обработки и валидации введенных пользователем данных, обработку исключительных ситуаций и корректное управление сеансами пользователей. Это позволяет обеспечить безопасность и надежность механизмов аутентификации и авторизации в приложении.

Отладка окна авторизации с применением класса шифрования (см. Рисунок 11).

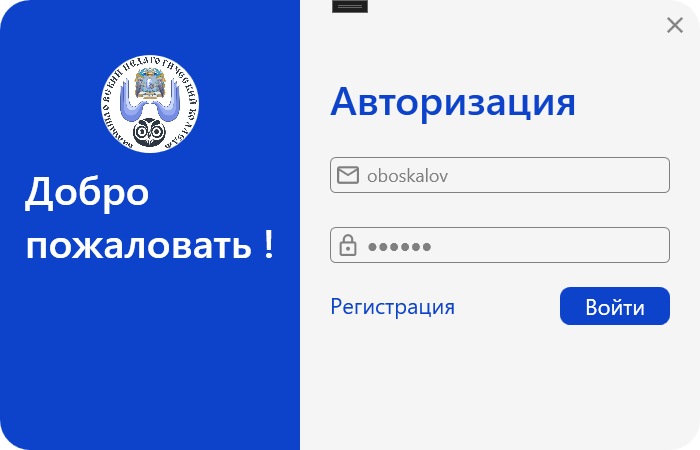


Рисунок 11 – Окно авторизация

Поставил точку остановы в коде (см. Рисунок 12).



Рисунок 12 - Точка остановы

Запустил программу, ввел данные и нажал кнопку «Войти», после чего перекинуло на код, в котором стояла точка остановы (см. Рисунок 13).

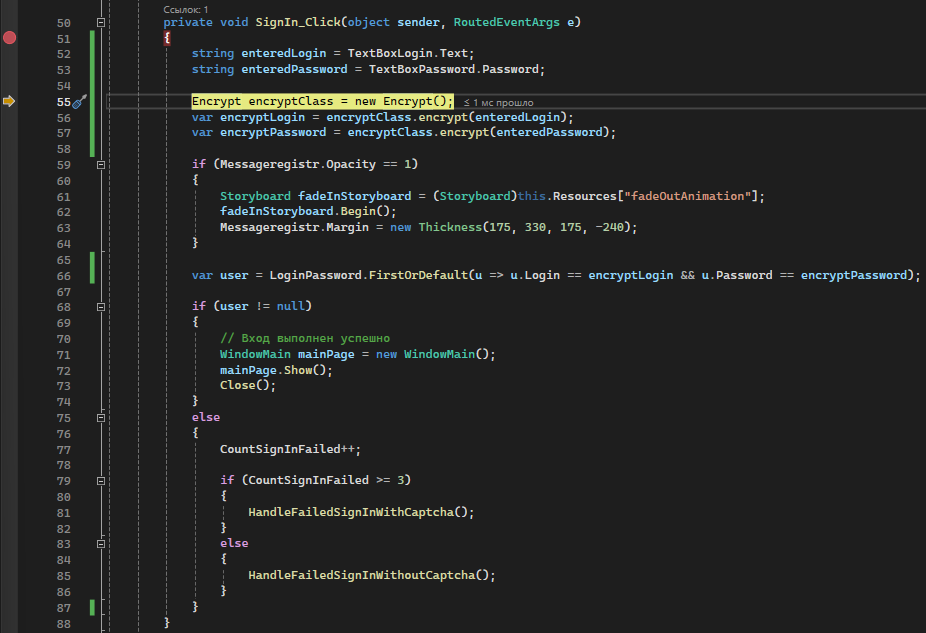


Рисунок 13 - Пошаговая отладка

После использования класса шифрования, вывел значения в окно контрольных значений (см. Рисунок 14).

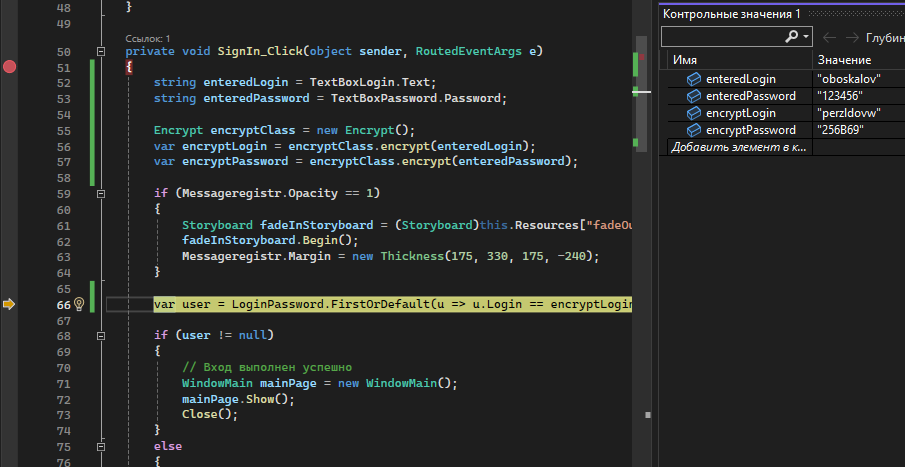


Рисунок 14 - Зашифрованный логин

После использования класса шифрования и установки точки останова в соответствующем месте кода, я убедился, что программа корректно принимает значения из полей логина и пароля. Затем, с помощью класса шифрования, введенные пользователем данные были зашифрованы. Для проверки корректности работы шифрования и сравнения значений с базой данных, были выведены значения в окно контрольных значений.

После ввода данных и нажатия кнопки "Войти", выполнение программы остановилось в месте точки останова, что позволило внимательно проанализировать текущее состояние и значения переменных в этот момент времени. Проверка значений, выведенных в окно контрольных значений, позволила убедиться в корректности работы класса шифрования и процесса сравнения зашифрованных данных с данными из базы данных.

Таким образом, успешное выполнение этапов отладки окна авторизации (см. Рисунок 15) с использованием класса шифрования подтвердило правильность его реализации и работу механизмов аутентификации в приложении.

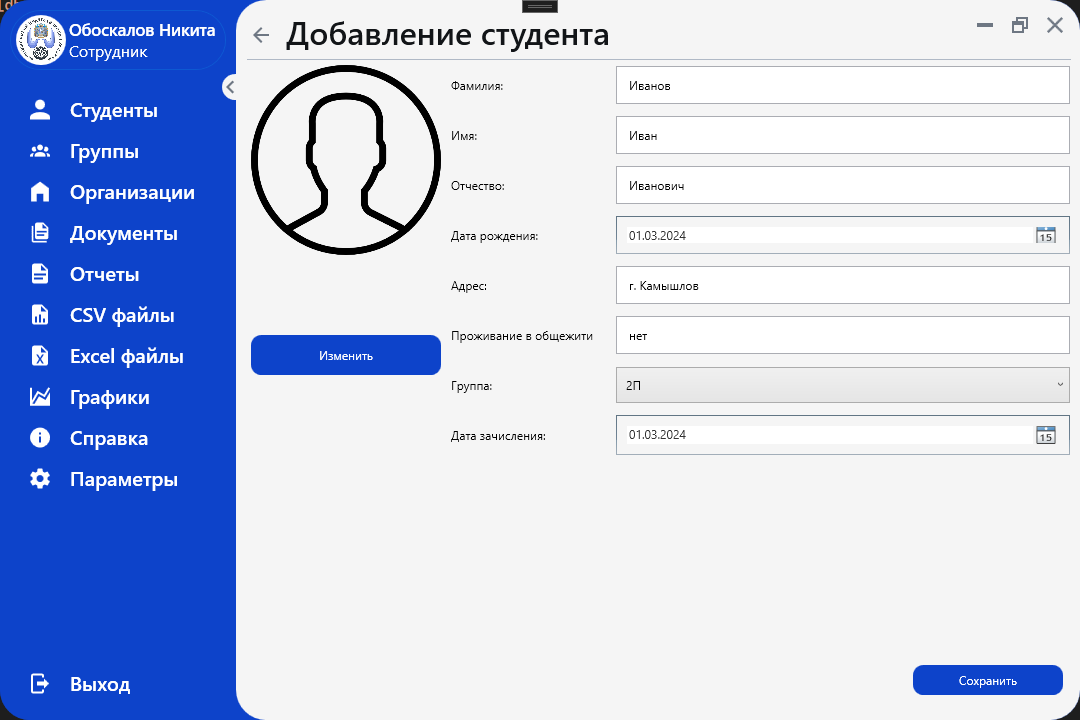


Рисунок 15 - Окно добавления студента

Протестировал страницу добавления студента и страницу редактирования группы по классам эквивалентности входных данных в соответствии с требованиями заказчика, используя различные ограничения при вводе данных (см. Таблицу 5).

Таблица 3 - Классы эквивалентности входных данных страницы добавление студента

| Входные условия | Классы эквивалентности | |
| --- | --- | --- |
| Правильные | Неправильные |
| Значение поля «Фамилия» на странице добавление студента | Текстовое значение от 1 до 250 символов (1) | -не введено (2)  -превышен лимит символов (3) |
| Значение поля «Имя» на странице добавление студента | Текстовое значение от 1 до 250 символов (4) | -не введено (5)  -превышен лимит символов (6) |
| Значение поля «Отчество» на странице добавление студента | Текстовое значение от 1 до 250 символов (7) | -не введено (8)  -превышен лимит символов (9) |
| Значение поля «Дата рождения» на странице добавление студента | Значение с 1.1.1900 по сегодняшнее (10) | -не выбрано (11)  -будущая дата (12) |
| Значение поля «Проживание в общежитии» на странице добавление студента | Значения “да” или “нет” (13) | -не введено (14)  -введены другие значения (15) |
| Значение поля «Группа» на странице добавление студента | Текстовое значение от 1 до 10 символов (19) | -не введено (20)  -превышен лимит символов (21) |
| Значение поля «Дата зачисления» на странице добавление студента | Значение с 1.1.1900 по сегодняшнее (10) | -не выбрано (11)  -будущая дата (12) |

Тест для правильных классов эквивалентности

Заполнение полей страницы добавление студента правильными значениями (охватывает классы 1, 3, 6, 9, 12, 15, 19, 22, 25, 28, 31, 34):

Фамилия: Иванов

Имя: Иван

Отчество: Иванович

Дата рождения: 01.03.2024

Адрес: г. Камышлов

Проживание в общежитие: нет

Группа: 2П

Дата зачисления: 01.03.2024

Результаты: при нажатии кнопки «Сохранить» введенный студент добавится в базу данных и будет выведено в окно для просмотра, при открытии диалогового окна с возможностью выбора студента будет возможность выбора только что добавленного ученика.

Тесты для неправильных классов эквивалентности

Заполнение полей страницы добавление студента неправильными значениями (охватывает классы 2, 4, 7, 10, 11, 13, 14, 16, 17, 18, 20, 21):

Фамилия: Иванов

Имя: Иван

Отчество: Иванович

Дата рождения: 01.03.2024

Адрес: г. Камышлов

Проживание в общежитие: нет

Группа: 2П

Дата зачисления: 01.03.2024

Результаты: при нажатии на кнопку «Сохранить» выведется сообщение об ошибке «Заполните все поля». При закрытии программы или обновлении страницы все введенные данные теряются.

При тестировании страницы добавления студента по классам эквивалентности входных данных в соответствии с требованиями заказчика были применены различные ограничения при вводе данных. Это включало в себя проверку на минимальную и максимальную длину полей, а также наличие специфических символов или форматов данных, если они были указаны в требованиях. Кроме того, осуществлялось тестирование на корректное обработку ошибок при вводе некорректных данных, таких как пустые поля или недопустимые символы. В результате тестирования было установлено соответствие функционала страницы требованиям заказчика, а также выявлены и исправлены потенциальные уязвимости или ошибки в обработке данных.

**Разработка сопровождающей технической документации**

Сопроводительная техническая документация представляет собой набор инструкций и руководств для пользователя программного обеспечения. Она может быть представлена в печатном или электронном виде и содержать различную информацию, такую как описание функций программы, инструкции по ее использованию и рекомендации по безопасному использованию.

После создания технической документации для автоматизированной информационной системы, было принято решение интегрировать руководство пользователя непосредственно в приложение. Это руководство содержит подробную информацию о функциях программы и правилах ее использования.

Чтобы воспользоваться руководством, пользователю необходимо перейти в меню «Справка» и выбрать пункт «Посмотреть руководство по использованию». После этого откроется диалоговое окно с руководством, в котором пользователь может найти необходимую информацию о работе с программой. Все разделы руководства подробно описывают соответствующие функции и дают практические рекомендации по их использованию.

**Оценка экономической/социальной эффективности проекта**

**Единовременные затраты**

Единовременные (капитальные) затраты вычисляются по следующей формуле:

К = Кпр+Ктс+Клс+Кпо+Кио+Коб+Коэ

где Кпр – затраты на проектирование;

Ктс – затраты на технические средства управления;

Клс – затраты на создание линий связи локальных сетей;

Кпо – затраты на программные средства;

Кио – затраты на формирование информационной базы;

Коб – затраты на обучение персонала;

Коэ – затраты на опытную эксплуатацию.

Затраты на проектирование рассчитываются следующим образом:

Кпр=Кзп+Кипс+Ксвт+Кпроч

где Кзп – затраты на заработную плату проектировщиков;

Кипс – затраты на инструментальные программные средства для проектирования;

Ксвт – затраты на средства вычислительной техники для проектирования;

Кпроч – прочие затраты на проектирование.

В рамках разработки дипломного проекта единовременные затраты сведены к нулю, так как разработка автоматизированной информационной системы не подразумевает какие-либо дополнительные покупки.

Так как для разработки автоматизированной информационной системы использовались инструментальные программные средства, то были выполнены соответствующие затраты для проектирования (см. Таблицу 6).

Таблица 6 – Затраты на инструментальные программные средства для проектирования.

| Программное средство | Стоимость |
| --- | --- |
| 1. Microsoft Visual Studio Community 2019 Версия 16.8.5 © Корпорация Майкрософт (Microsoft Corporation), 2020 | 0,00 ₽ |
| 2. SQL Server Management Studio v18.8 © Корпорация Майкрософт (Microsoft Corporation), 2020 | 0,00 ₽ |
| 3. Microsoft® Office 2016 MSO (16.0.4266.1001), 64-разрядная версия. | 0,00 ₽ |
| Итого: | 0,00 ₽ |

Прочие затраты на проектирование составляют 0% от совокупных затрат на проектирование.

Полная стоимость затрат на проектирование составляет:

Кпр=з/пл+затр на инстр ПС +прочие затраты

Затраты на технические средства управления составляют 0 руб, так как нет необходимости в их приобретении. Амортизационные отчисления за время проектирования очень малы.

Затраты на формирование информационной базы включают в себя:

Заработную плату администратора, программное обеспечение для работы с базой данных и затраты на ее хранение. Стоимость хранения базы данных равняется нулю так как база данных будет использоваться лишь в локальной сети и не нуждается в хостинге, поэтому в данных затратах не берется в расчет. Программное обеспечение, которое использовалось для работы с базой данных, является бесплатным (см. Таблицу 7).

Затраты на обучение персонала составляют 0 = з/п / кол-во раб дней = 0 руб. В рамках разработки дипломного проекта затраты на обучение персонала не предоставляются.

Затраты на опытную эксплуатацию сведены к нулю, так как в этом нет необходимости.

Таблица 7 – Единовременные и прочие затраты

| Вид затрат | Капитальные затраты | |
| --- | --- | --- |
| Состав затрат | Планируемая сумма |
| Единовременные затраты | Microsoft Visual Studio Community 2019 Версия 16.8.5 © Корпорация Майкрософт (Microsoft Corporation), 2020 | 0,00 ₽ |
| SQL Server Management Studio v18.8 © Корпорация Майкрософт (Microsoft Corporation), 2020 | 0,00 ₽ |
| Microsoft® Office 2016 MSO (16.0.4266.1001), 64-разрядная версия. | 0,00 ₽ |
| Прочие затраты | Затраты на обучение персонала | 0,00 ₽ |
| Затраты на опытную эксплуатацию | 0,00 ₽ |

Оценка внедрения: разработка является наиболее эффективной с экономической точки зрения так как обслуживание и внедрение данной разработки не требует дополнительных затрат кроме программных средств и заработной платы администратора, работающего с автоматизированной информационной системой.

Ключевыми факторами социальной и экономической эффективности являются:

1. Минимизация упущенного дохода или формирование новых источников дохода;
2. Снижение текущих производственных (эксплуатационных) затрат;
3. Снижение административно-управленческих затрат;
4. Минимизация налоговых и других обязательных выплат;
5. Снижение потребности в капитальных затратах;
6. Увеличение оборачиваемости текущих активов.

Возможные статьи дохода от внедрения разработанного программного продукта в ГАПОУ СО «Камышловский педагогический колледж»:

* Увеличение производительности работы медицинского кабинета колледжа: Разработанный программный продукт может улучшить эффективность ведения медицинской документации, позволив быстрее и точнее регистрировать данные студентов, отслеживать медицинские аспекты и проводить анализ состояния здоровья студентов.
* Снижение затрат на медицинские услуги и лекарственные средства: благодаря более эффективному мониторингу состояния здоровья студентов и своевременной диагностике возможных проблем, колледж может снизить затраты на медицинские услуги и лекарственные средства за счет предотвращения возможных заболеваний или своевременного лечения.
* Оптимизация учебного процесса: Улучшение доступа к информации о состоянии здоровья студентов может способствовать более эффективной организации учебного процесса. Это может привести к повышению успеваемости студентов, сокращению времени на административные процедуры и увеличению общей эффективности образовательной деятельности.

Эффектом внедрения информационной системы (ИС) в организацию является улучшение эффективности ее деятельности, оптимизация процессов и повышение конкурентоспособности. Внедрение ИС может привести к следующим результатам:

1.Автоматизация и оптимизация бизнес-процессов: ИС позволяет автоматизировать рутинные операции, сокращает время выполнения задач, улучшает координацию работы между подразделениями и сокращает вероятность ошибок.

2.Улучшение доступа к информации: ИС обеспечивает быстрый и удобный доступ к необходимой информации для принятия решений на всех уровнях управления. Это позволяет сократить время на поиск и обработку данных.

3.Экономия времени и ресурсов: ИС помогает оптимизировать расходы на персонал, сокращает затраты на бумажную документацию, минимизирует потери информации и уменьшает количество ошибок.

4.Улучшение управленческих решений: ИС предоставляет руководителям точную и своевременную информацию для анализа ситуации и принятия обоснованных управленческих решений.

Существуют два типа методов оценки эффективности проекта: финансовый и качественный. В данной работе оценка будет производиться по методу информационной экономики (InformationEconomics), которая относится к качественным методам.

Следуя данному методу, в первую очередь необходимо определить цели, которые нужно достичь с помощью внедрения проекта:

* Повышение производительности: В контексте медицинского кабинета колледжа это может означать увеличение скорости обработки медицинских данных, сокращение времени на административные процедуры, повышение эффективности работы персонала и оптимизацию использования ресурсов.
* Сокращение времени выполнения административных процедур: Целью может быть сокращение времени, затрачиваемого на оформление документов, ведение отчетности и другие административные процедуры, связанные с медицинским кабинетом колледжа. Это поможет освободить время персонала для более важных задач и улучшить общую эффективность работы.
* Улучшение доступности и точности медицинских данных: Целью может быть обеспечение легкого доступа к медицинской информации о студентах для врачей и других уполномоченных лиц, а также повышение точности и достоверности этой информации. Это поможет улучшить качество медицинского обслуживания студентов и снизить риск ошибок.
* Увеличение безопасности и конфиденциальности данных: Целью может быть обеспечение надежной защиты медицинских данных студентов от несанкционированного доступа и утечек информации. Это поможет соблюдать требования по защите конфиденциальности и приватности данных, а также повысит доверие студентов к колледжу.
* Сокращение рисков и повышение ответственности: Целью может быть сокращение рисков возникновения медицинских ошибок, неправильного ведения медицинской документации и других негативных событий. Повышение ответственности перед студентами и их родителями за состояние здоровья и безопасность также может быть важной целью внедрения информационной системы.

Реализованный программный продукт удовлетворяет всем поставленным целям.

Перечень возможных рисков проекта:

* Технические риски: Проблемы совместимости с существующими информационными системами колледжа. Трудности при интеграции новой информационной системы с другими учебными или административными системами.
* Безопасностные риски: Угрозы кибербезопасности, такие как хакерские атаки или вирусы, которые могут угрожать конфиденциальности медицинских данных студентов. Недостаточная защита данных от несанкционированного доступа или утечки информации.
* Организационные риски: Сопротивление персонала изменениям или нехватка поддержки со стороны управления колледжа. Недостаточная подготовка персонала к использованию новой информационной системы.
* Финансовые риски: Превышение бюджета проекта из-за непредвиденных затрат или недостаточной оценки стоимости реализации. Неэффективное использование ресурсов или перерасход средств на проекте.
* Технологические риски: Недостаточная надежность или производительность используемого программного обеспечения. Отсутствие поддержки или обновлений для выбранного программного обеспечения в будущем.
* Людские риски: Уход ключевых специалистов, работающих над проектом, что может привести к задержкам в разработке или потере ценной экспертизы. Недостаточная коммуникация и сотрудничество между участниками проекта, что может замедлить процесс реализации.

Провел качественную оценку рисков. Риски оцениваются по трехуровневой шкале влияния рисков на реализацию проекта:

– «низкий уровень» – малый риск, незначительное влияние на проект;

– «средний уровень» – риск, имеющий равную вероятность повлиять и не повлиять на результат исполнения проекта;

– «высокий уровень» – высокий риск, заключающийся в неисполнении проекта или в значительном изменении результата его реализации.

Результаты качественной оценки рисков приведены далее в таблице 8. Также в таблице указаны мероприятия по снижению вероятности рисков, имеющих наивысшую оценку.

Таблица 8 – Риски проекта

| Риск | Потери | Уровень влияния риска на проект | Возможность предотвращения или снижения риска |
| --- | --- | --- | --- |
| Недостаточная квалификация | Ненамеренное повреждение файлов сотрудниками организации | «средний уровень» | Самообразование |
| Недостаточная защита информационных ресурсов | Утечка конфиденциальной информации | «высокий уровень» | Предотвращение утечки, хищения, утраты, искажения, подделки информации |
| взлом веб-приложения злоумышленниками; |
| Сбои в работе системы | Повышенная нагрузка, ведущая к снижению результатов. | «высокий уровень» | Перезапуск автоматизированной информационной системы |

Объектом проведенного исследования является рабочее место сотрудника медицинского кабинета ГАПОУ СО «Камышловский педагогический колледж».

Преимущества использования специалистом разработанного продукта заключаются в:

* Снижение затрат: Программный продукт, разработанный студентами, часто может быть доступен по более низкой цене или даже бесплатно, что снижает затраты на приобретение программного обеспечения для организации.
* Инновационные решения: Студенты, работающие над дипломными проектами, часто вносят свежие идеи и инновационные решения, которые могут быть полезны для организации. Использование продукта, созданного студентами, может способствовать внедрению новых подходов и методов в работу организации.
* Поддержка образовательных инициатив: Поддержка программ дипломной практики и научно-исследовательских проектов студентов способствует развитию образовательной среды и сотрудничеству между образовательными учреждениями и бизнесом.
* Адаптивность и гибкость: Программный продукт, созданный студентами, может быть более адаптивным и гибким к потребностям организации, так как он может разрабатываться с учетом специфики ее бизнес-процессов и требований.
* Развитие потенциала персонала: Участие в разработке программного продукта во время дипломной практики может быть ценным опытом для студентов и способствовать их профессиональному развитию. В то же время, использование такого продукта позволяет организации развивать потенциал молодых специалистов и привлекать их в будущем на работу.

Риски при использовании продукта заключаются в следующем:

* Недостаточное качество продукта: Программный продукт может оказаться низкого качества из-за ошибок в разработке, несоответствия требованиям пользователя или недостаточного тестирования перед внедрением.
* Проблемы совместимости: Продукт может оказаться несовместимым с другими программными системами или оборудованием, что может затруднить его интеграцию в существующую инфраструктуру организации.
* Отсутствие поддержки и обновлений: если разработчики продукта не обеспечивают его поддержку и регулярные обновления, это может привести к проблемам с безопасностью, исправлением ошибок и адаптацией к изменяющимся потребностям организации.
* Недостаточная защита данных: Программный продукт может оказаться уязвимым к кибератакам или не обеспечивать достаточной защиты данных, что может привести к утечкам информации или потере конфиденциальности.
* Неэффективное использование: если пользователи не будут правильно обучены или не увидят ценности в использовании программного продукта, это может привести к его неполному или неэффективному использованию, что снизит общую эффективность и рентабельность внедрения.
* Недостаточная гибкость и адаптивность: Программный продукт может оказаться недостаточно гибким или неспособным адаптироваться к изменяющимся потребностям организации или бизнес-процессам, что снизит его ценность на длительную перспективу.

С учетом вышеописанных преимуществ, использование разработанного продукта является эффективным и обоснованным. Продукт обладает рядом преимуществ, которые способствуют повышению эффективности работы организации и достижению ее целей.

Выводы по разделу «Экономическая и социальная эффективность от внедрения.

В данном разделе произведен расчет всех капитальных (единовременных) и эксплуатационных затрат, а также совокупной стоимости владения информационной системой (ИС).

Далее произведена оценка внедрения ИС как стратегического проекта, в рамках которого учитываются его влияние на бизнес-процессы, цели и стратегические планы организации.

Также составлен перечень возможных рисков проекта, проведена качественная оценка и указаны мероприятия по снижению вероятности рисков и их влияния на успешное внедрение и эксплуатацию ИС.

Все полученные данные представлены в сводных таблицах и диаграммах для наглядного и комплексного анализа и принятия решений о дальнейших шагах в реализации проекта.

**Заключение**

**Список литературы**

1. Арлоу Д., Нейштадт И. UML 2 и Унифицированный процесс. Практический объектноориентирован ный анализ и проектирование, 2е издание. – Пер. с англ. – СПб: Символ Плюс, 2007. – 624 с., ил.
2. Бахтизин В.В. Методология функционального проектирования IDEF0: Учеб. пособие по курсу «Технология разработки программного обеспечения» для студ. спец. 40 01 01 “Программное обеспечение информационных технологий дневной формы обучения” В.В.Бахтизин, Л.А.Глухова. – Мн.: БГУИР, 2003. – 24 с.: ил.
3. Бурцева Е.В. Информационные системы : учеб. пособие /, И.П. Рак, А.В. Селезнев, А.В. Терехов, В.Н. Чернышов. – Тамбов : Изд-во Тамб. гос. техн. ун-та, 2009. – 128 с. – 150 экз. – ISBN 978-5-8265-0874-9.
4. Васильков, А. В. Безопасность и управление доступом в информационных системах / А.В. Васильков, И.А. Васильков. - М.: Форум, 2015. - 368 c.
5. Горбаченко В. И. Проектирование информационных систем с CA ERwin Modeling Suite 7.3 : учебное пособие / В. И. Горбаченко, Г. Ф. Убиенных, Г. В. Бобрышева – Пенза: Изд-во ПГУ, 2012. – 154 с.
6. Горлушкина Н.Н. Системный анализ и моделирование информационных процессов и систем. – СПб: Университет ИТМО, 2016. – 120 с.
7. ГОСТ 19.104-78 Единая система программной документации (ЕСПД). Основные надписи (с Изменением N 1) от 18 декабря 1978 г. N 3351 дата введения установлена 01.01.80 ИЗДАНИЕ (январь 2010 г.) с Изменением N 1, утвержденным в сентябре 1981 г., Пост. N 4150 от 07.09.81 (ИУС 11-81).
8. ГОСТ 19.105-78 Единая система программной документации (ЕСПД). Общие требования к программным документам (с Изменением N 1) от 18 декабря 1978 г. N 3350 дата введения установлена 01.01.80 ИЗДАНИЕ (январь 2010 г.) с Изменением N 1, утвержденным в сентябре 1981 г. (ИУС 11-81).
9. ГОСТ 19.106-78 Единая система программной документации (ЕСПД). Требования к программным документам, выполненным печатным способом (с Изменением N 1) от 18 декабря 1978 г. N 3350 дата введения установлена 01.01.80 ИЗДАНИЕ (январь 2010 г.) с Изменением N 1, утвержденным в сентябре 1981 г. (ИУС 11-81).
10. ГОСТ 19.201-78 Единая система программной документации (ЕСПД). Техническое задание. Требования к содержанию и оформлению (с Изменением N 1) от 18 декабря 1978 N 3351 дата введения установлена 01.01.80 ИЗДАНИЕ (январь 2010 г.) с Изменением N 1, утвержденным в июне 1981 г. (ИУС 9-81).
11. ГОСТ 2.301-68 Единая система конструкторской документации (ЕСКД). Форматы (с Изменениями N 1, 2, 3) от 28 мая 1968 ИЗДАНИЕ (август 2007 г.) с Изменениями N 1, 2, 3, утвержденными в декабре 1980 г., марте 1989 г., июне 2006 г. (ИУС 3-81, 7-89, 9-2006).
12. Грекул, В. И. Проектирование информационных систем : учебник и практикум для среднего профессионального образования / В. И. Грекул, Н. Л. Коровкина, Г. А. Левочкина. — Москва : Издательство Юрайт, 2022. — 385 с. — (Профессиональное образование). — ISBN 978-5-534-12104-9. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/496196> (дата обращения: 09.03.2022).
13. Зараменских, Е. П. Информационные системы: управление жизненным циклом : учебник и практикум для среднего профессионального образования / Е. П. Зараменских. — Москва : Издательство Юрайт, 2022. — 431 с. — (Профессиональное образование). — ISBN 978-5-534-11624-3. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/495987> (дата обращения: 09.03.2022).
14. Иванов Д. Ю., Новиков Ф. А. Основы моделирования на UML: Учеб. пособие. – СПб.: Изд-во Политехн. ун-та, 2010. – 249с.
15. Информационные системы - миф и действительность. - М.: Знание, 2016. - 887 c.
16. Кара-Ушанов В.Ю. SQL — язык реляционных баз данных : учебное пособие / В.Ю. Кара-Ушанов.— Екатеринбург : Изд-во Урал. ун-та, 2016.— 156 с.
17. Карпова И.П Базы данных. Курс лекций и материалы для практических заданий. – Учебное пособие. – М.: Питер, 2013. – 240 с.
18. Масленникова, О.Е. Разработка реляционных баз данных с использованием CASE-средства ALL Fusion Data Modeler [Электронный ресурс] / О.Е. Масленникова, О.Б. Назарова. - М. : ФЛИНТА, 2016. - 74 с. - URL: http://e.lanbook.com/ (ЭБС «ЛАНЬ») (дата обращения: 11.03.2022).
19. Маторин С.И. Теория систем и системный анализ: Учебное пособие / С.И. Маторин., О.А. Зимовец. – Белгород: Изд-во НИУ «БелГУ», 2012. – 288 с.
20. Медведев М. А. Программирование на СИ# : учеб. пособие / М. А. Медведев, А.Н. Медведев. — Екатеринбург : Изд-во Урал. ун-та, 2015. — 64 с.
21. Мезенцев, К. Н. Автоматизированные информационные системы / К.Н. Мезенцев. - М.: Academia, 2016. - 176 c.
22. Осипов Д. Л. Технологии проектирования баз данных. – М.: ДМК Пресс, 2019. – 498 с.: ил.
23. Попова-Коварцева, Дарья Александровна Основы проектирования баз данных: учеб. пособие / Д.А. Попова-Коварцева, Е.В. Сопченко. – Самара: Изд-во Самарского университета, 2019. – 112 с.: ил.
24. Райордан Р. Основы реляционных баз данных/Пер, с англ. — М.: Издательско-торговый дом «Русская Редакция», 2001. — 384 с.: ил.
25. Ракитов А.И. Системный анализ и аналитические исследования: руководство для профессиональных аналитиков / А.И. Ракитов, Д.А. Бондяев, И.Б. Романов, СВ. Егерев, А.Ю. Щербаков; [ОТВ. ред. А.И. Ракитов]. - М., 2009. - 448 с
26. Редько, В.Н. Базы данных и информационные системы / В.Н. Редько, И.А. Басараб. - М.: Знание, 2016. - 667 c.
27. Сергеев С. Ф., Падерно П. И., Назаренко Н. А. Введение в проектирование интеллектуальных интерфейсов: Учебное пособие. - СПб: СПбГУ ИТМО, 2011. - 108 с.
28. *Сергеев, А. А.* Бизнес-планирование : учебник и практикум для среднего профессионального образования / А. А. Сергеев. — 4-е изд., испр. и доп. — Москва: Издательство Юрайт, 2022. — 484 с. — (Профессиональное образование). — ISBN 978-5-534-14036-1. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/495183> (дата обращения: 10.03.2022).
29. Сибилев В.Д. Модели и проектирование баз данных [Электронный ресурс] URL: <https://www.studmed.ru/sibilev-vd-modeli-i-proektirovanie-baz-dannyh_dbf86c7f1ce.html> (дата обращения: 12.03.2022).
30. Структурный анализ потоков данных (Data Flow Diagrams – DFD): метод. указания / сост. Д.Ю. Киселев, Ю.В. Киселев, В.Д. Макарьев. – Самара: Изд-во СГАУ, 2014. – 12 с.
31. Фаулер M. UML. Основы, 3е издание. – Пер. с англ. – СПб: СимволПлюс, 2004. – 192 с., ил.
32. Федорова Г. Н. Информационные системы: учебник для студ. учреждений сред. проф. образования / Г. Н. Федорова. — 3-е изд., стер. — М. : Издательский центр «Академия», 2013. — 208 с.
33. Фуфаев Э. В. Базы данных : учеб. пособие дли студ. учреждений сред, проф. образования / Э. В.Фуфаев, Д. Э.Фуфаев. — 7-е изд., стер. — М .: Издательский центр «Академия», 2012. — 320 с.
34. Хайруллин Р.С. Программирование на C#: учебное пособие. – Хайруллин Р.С. – Казань: Изд-во Казан.гос.архитект.-строит.ун-та, 2017. – 159 с.
35. Халимон, В.И. Базы данных: учебное пособие / В.И. Халимон, Г.А. Мамаева, А.Ю. Рогов, В.Н. Чепикова - С-Пб.: СПбГТИ(ТУ), 2017. – 118 с.

**Программный продукт**

**Приложения**

**Приложение 1**

**Глоссарий сущностей**

| **Имя сущности** | **Описание сущности** |
| --- | --- |
| USER\_ROLE (Роль пользователя) | Роль пользователя в приложении, определяет к каким данным и функциям будет доступ |
| USER (Пользователь) | Лицо, которое будет пользоваться программой |
| CHECK\_VACCINE (Осмотры перед прививками и после прививок) | Документ “Осмотры перед прививками и после прививок” |
| DATA\_MEDICAL\_CHECKUPS (Данные медицинских профилактических осмотров) | Документ “Данные медицинских профилактических осмотров” |
| GROUP (Группа) | Группа студентов, которые проходят обучении в колледже |
| MEDICAL\_ORGANIZATION (Медицинская организация) | Медицинская организация - юридическое лицо или индивидуальный предприниматель, осуществляющий медицинскую деятельность и имеющий на это соответствующую лицензию. |
| OPERATIONS (Операции) | Операции студента за период обучения в колледже |
| PASSES\_FOR\_ILLNESS (Пропуски по болезни) | Пропуски занятий по болезни |
| PREVENTIVE\_VACCINE (Профилактические прививки) | Профилактические прививки студента |
| PASS\_CONROLLER (Ответственный за пропуски) | Студент ответственный за пропуски занятий в группе |
| SANITARY\_SPA\_TREATMENT (Санитарно-курортное лечение) | Санитарно-курортное лечение студента |
| HOSPITAL\_TREATMENT (Стационарное лечение) | Стационарное лечение студента |
| LOGIN\_PASSWORD (Логины и пароли) | Таблица содержит логины и пароли пользователей |
| SURVEYS (Обследования) | Обследования студента за период обучения в колледже |
| PAST\_INFECTIOUS\_DIASES (Перенесенные инфекционные заболевания) | Перенесенные студентом инфекционные заболевания |
| TRAUMA (Травмы) | Травмы студента за период обучения в колледже |

**Приложение 1**

**Глоссарий атрибутов**

| **Имя сущности** | **Имя атрибута** | **Описание атрибута** |
| --- | --- | --- |
| USER\_ROLE (Роль пользователя) | ID\_USER\_ROLE | Идентификационный номер роли пользователя |
|  | NameRole | Наименование роли |
| USER (Пользователь) | ID\_USER | Идентификационный номер пользователя |
|  | ID\_USER\_ROLE | Роль пользователя в приложении |
|  | Surname | Фамилия |
|  | Name | Имя |
|  | Patronymic | Отчество |
|  | DateBirth | Дата рождения |
|  | HomeAddress | Домашний адрес |
|  | Hostel | Проживает ли пользователь в общежитие |
|  | ID\_GROUP | Идентификационный номер группы |
|  | DateEnrollment | Дата поступления |
|  | Color | Цвет пользователя |
|  | ImageName | Фото пользователя |
| CHECK\_VACCINE (Осмотры перед прививками и после прививок) | ID\_CHECK\_VACCINE | Идентификационный номер документа “Осмотры перед прививками и после прививок” |
|  | ID\_USER | Идентификационный номер пользователя |
|  | Date | Дата |
|  | Body\_temp | Температура тела |
|  | ResultInspecion | Результат осмотра |
|  | Vaccine | Прививка |
|  | Permission | Разрешена или не разрешена |
|  | StateAfterThirtyMin | Состояние через 30 мин |
| DATA\_MEDICAL\_CHECKUPS (Данные медицинских профилактических осмотров) | ID\_DATA\_MEDICAL\_CHECKUPS | Идентификационный номер документа “Данные медицинских профилактических осмотров” |
|  | ID\_USER | Идентификационный номер пользователя |
|  | ID\_MEDICAL\_ORGANIZATION | Идентификационный номер медицинской организации |
|  | Year | Год |
|  | Age | Возраст |
|  | Survey | Обследование |
|  | Survey\_result | Результат обследования |
| GROUP (Группа) | ID\_GROUP | Идентификационный номер группы |
|  | Name\_group | Имя группы |
|  | Number\_of\_persons | Кол-во человек |
| MEDICAL\_ORGANIZATION (Медицинская организация) | ID\_MEDICAL\_ORGANIZATION | Идентификационный номер медицинской организации |
|  | Name\_medical\_organization | Название медицинской организации |
|  | Adres | Адрес |
| OPERATIONS (Операции) | ID\_OPERATIONS | Идентификационный номер документа “Операции” |
|  | ID\_USER | Идентификационный номер пользователя |
|  | Operations |  |
| PASSES\_FOR\_ILLNESS (Пропуски по болезни) | ID\_PREVENTIVE\_VACCINE | Идентификационный номер документа “Пропуски по болезни” |
|  | ID\_USER | Идентификационный номер пользователя |
|  | DateFrom | Дата начала |
|  | DateTo | Дата конца |
|  | Diagnosis | Диагноз |
| PREVENTIVE\_VACCINE (Профилактические прививки) | ID\_PREVENTIVE\_VACCINE | Идентификационный номер документа “Профилактические прививки” |
|  | ID\_USER | Идентификационный номер пользователя |
|  | ID\_MEDICAL\_ORGANIZATION | Идентификационный номер медицинской организации |
|  | Vaccine | Прививка |
|  | Date | Дата когда поставили прививку |
| PASS\_CONROLLER (Ответственный за пропуски) | ID\_PASS\_CONROLLER | Идентификационный номер ответственного за пропуски |
|  | ID\_USER | Идентификационный номер пользователя |
| SANITARY\_SPA\_TREATMENT (Санитарно-курортное лечение) | ID\_SANITARY\_SPA\_TREATMENT | Идентификационный номер документа “Санитарно курортного лечения” |
|  | ID\_USER | Идентификационный номер пользователя |
|  | Description | Описание |
| HOSPITAL\_TREATMENT (Стационарное лечение) | ID\_STATS\_TREATMENT | Идентификационный номер пользователя |
|  | ID\_USER | Идентификационный номер пользователя |
|  | Description | Описание |
| SURVEYS (Обследования) | ID\_SURVEYS | Идентификационный номер обследования |
|  | ID\_USER | Идентификационный номер пользователя |
|  | Year | Год |
|  | Survey | Обследование |
|  | Survey\_result | Результат обследования |
|  | ID\_MEDICAL\_ORGANIZATION | Идентификационный номер медицинской организации |
| PAST\_INFECTIOUS\_DIASES (Перенесенные инфекционные заболевания) | ID\_PAST\_INFECTIOUS\_DIASES | Идентификационный номер документа “Перенесенные инфекционные заболевания” |
|  | ID\_USER | Идентификационный номер пользователя |
|  | Date | Дата |
|  | Description | Заболевание |
| TRAUMA (Травмы) | ID\_TRAUMA | Идентификационный номер документа “Травмы” |
|  | ID\_USER | Идентификационный номер пользователя |
|  | Description | Описание |