

МИНИСТЕРСТВО ОБРАЗОВАНИЯ КИРОВСКОЙ ОБЛАСТИ
Кировское областное государственное профессиональное образовательное
бюджетное учреждение
"Слободской колледж педагогики и социальных отношений"

КУРСОВОЙ ПРОЕКТ

по ПМ 01 «Разработка программных модулей» на тему:
**РАЗРАБОТКА ПРОГРАММНОГО МОДУЛЯ ДЛЯ АВТОМАТИЗАЦИИ
УЧЕТА АНАЛИЗОВ МЕДИЦИНСКОЙ ЛАБОРАТОРИИ**

Выполнила: Коротких Наталья
Михайловна

Специальность 09.02.07
Информационные системы и
программирование

Группа 21П-1
Форма обучения: очная

Руководитель: Махнев
Александр Анатольевич

Дата защиты курсового проекта:

Председатель ПЦК:

Оценка за защиту курсового проекта:

Слободской
2024

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ	3
1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ	5
2. РАЗРАБОТКА ТЕХНИЧЕСКОГО ЗАДАНИЯ	9
3. ОПИСАНИЕ АЛГОРИТМОВ И ФУНКЦИОНИРОВАНИЯ ПРОГРАММЫ.....	12
4. ТЕСТИРОВАНИЕ ПРОГРАММНОГО МОДУЛЯ	22
5. РУКОВОДСТВО ОПЕРАТОРА	25
ЗАКЛЮЧЕНИЕ	30
СПИСОК ЛИТЕРАТУРЫ	31
ПРИЛОЖЕНИЯ	33

ВВЕДЕНИЕ

В условиях современного общества медицинские лаборатории занимают ключевое место в системе здравоохранения, обеспечивая высококачественную диагностику и мониторинг состояния здоровья пациентов. С увеличением объема проводимых исследований и ростом требований к скорости и качеству лабораторных анализов, а также с повышением уровня автоматизации в медицине перед лабораториями встают новые задачи. Эти изменения требуют от медицинских учреждений внедрения современных технологий и оптимизации процессов для обеспечения надежности и точности результатов.

Нерациональное управление процессами учета может привести к ошибкам в результатах, задержкам в проведении анализов и, как следствие, к снижению качества медицинского обслуживания. Важно отметить, что ошибки в лабораторной диагностике могут иметь серьёзные последствия для здоровья пациентов, что делает актуальной задачу оптимизации и автоматизации процессов в медицинских лабораториях. Автоматизация не только способствует повышению эффективности работы, но и снижает вероятность ошибок, что, в свою очередь, улучшает качество обслуживания пациентов.

Реализация данного модуля требует всестороннего анализа текущих бизнес-процессов медицинских лабораторий, а также глубокого понимания специфики работы в сфере лабораторной диагностики.

Объект исследования – процесс учета анализов (результатов исследований) медицинской лаборатории.

Предмет исследования — разработка программного модуля для автоматизации учета анализов (результатов исследований) медицинской лаборатории.

Цель курсового проекта – разработка программного модуля для повышения эффективности работы медицинской лаборатории, который автоматизирует учет и обработку данных, минимизирует вероятность ошибок

при анализе результатов исследований и улучшает качество обслуживания пациентов, обеспечивая при этом надежность и простоту использования.

Задачи:

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

Методы исследования:

- Провести анализ предметной области.
- Изучить существующие аналоги программ.
- Разработать и провести тестирование программного модуля.

В рамках работы будет проведен комплексный анализ предметной области, что позволит выявить ключевые аспекты, требующие автоматизации, а также определить функциональные требования к программному продукту.

В результате выполнения данного курсового проекта планируется создание эффективного и надежного программного модуля для автоматизации процессов учета анализов медицинской лаборатории. Этот модуль будет соответствовать требованиям современного рынка, обеспечивая высокое качество предоставляемых услуг.

1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

Медицинская лаборатория — это лаборатория, которая проводит биологические, микробиологические, иммунологические, химические, иммуногематологические и другие исследования материалов из организма человека в целях получения информации для диагностики, предупреждения и лечения болезни или оценки состояния здоровья человека и которая может оказать консультативную помощь относительно всех аспектов лабораторных исследований, включая интерпретацию результатов и рекомендацию дальнейших не обходимых исследований [3]. Для таких исследований используется биологический материал.

Биологический материал, в том числе кровь, в соответствии с ГОСТ Р 53079.4-2008 [2] — это жидкости, ткани и экскреты человека (спинномозговая жидкость, моча, слюна).

Большая часть клинических лабораторных исследований проводится в образцах крови: венозной, артериальной или капиллярной.

Биохимический анализ слюны — исследование, с помощью которого оцениваются показатели, отражающие состояние микрофлоры ротовой полости.

Сотрудники, которые задействованы в проведении исследований в медицинской лаборатории:

- медицинский регистратор (регистратор) — сотрудник поликлиники, работающий в регистратуре.

Он вносит данные о клиенте в систему, осуществляет распределение клиентов по специалистам и запись на прием, а также ведет картотеку организации.

- Медсестра осуществляет подготовку пациента и контроль его состояния во время процедуры, соблюдение санитарных норм, а также ведение служебной документации и может отметить в системе о приеме пациента, когда был взят биоматериал и сформировать Штрихового код (Штрих-кода) или QR-код для пробирки.

- Лаборант — это специалист, который занимается проведением лабораторных исследований, т. е. готовит препараты и приборы для исследований, а также подготавливает лабораторное оборудование и следит за его исправностью. Он принимает участие в проведении исследований, экспериментов, обеспечивает лабораторию необходимыми материалами, реактивами, записывает показания приборов [4].

После изучения основных понятий и функций сотрудников бала построена диаграмма вариантов использования (use case diagram).

На данной диаграмме выделены три ключевых действующих лица: лаборант, регистратор и медсестра. Это визуальное представление помогает лучше понять процессы внутри лаборатории и их взаимосвязь.

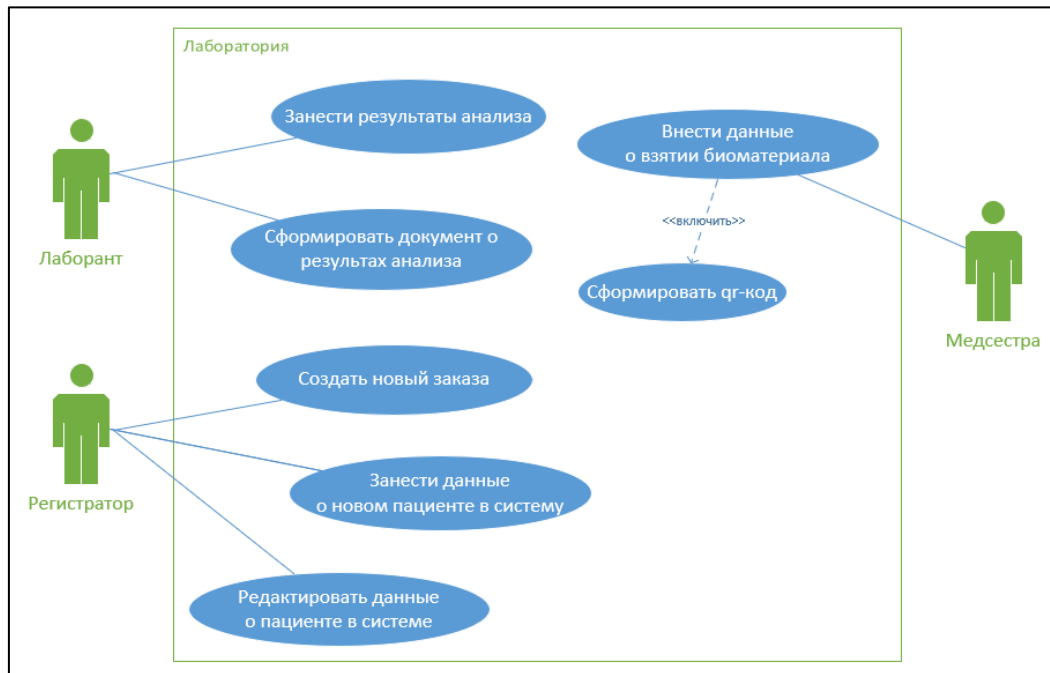


Рисунок 1 – Диаграмма вариантов использования

Следующим шагом стало создание ER-диаграммы предметной области, которая наглядно отражает структуру данных и их взаимосвязи. Эта диаграмма станет основой для проектирования базы данных, необходимой для автоматизации процессов учета анализов.

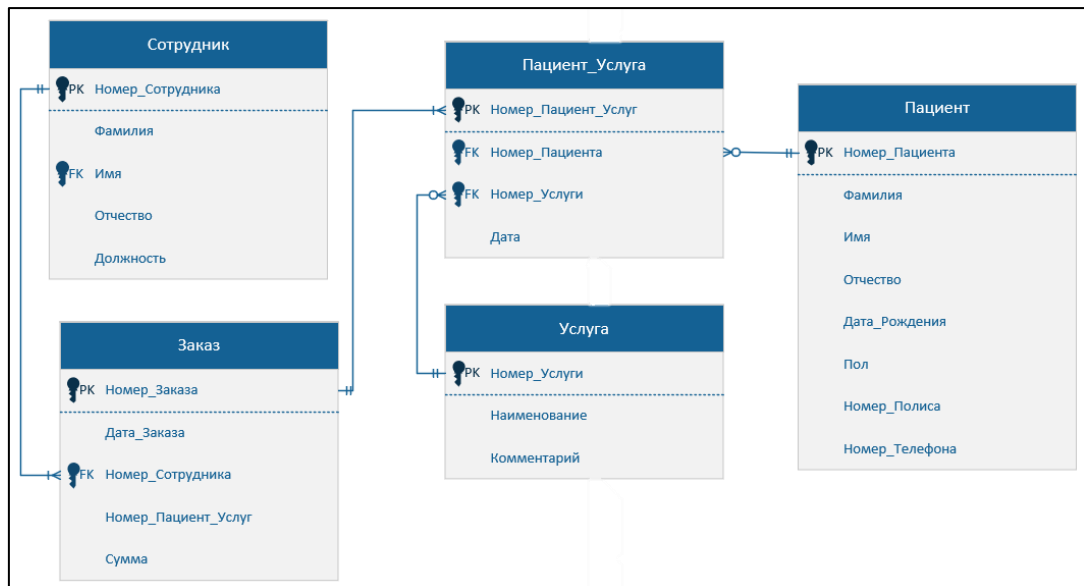


Рисунок 2 – ER-диаграмма предметной области

Понимание структуры данных является ключевым аспектом успешной реализации проекта. Однако для достижения наилучших результатов важно также изучить аналогичные системы на рынке.

Программы аналоги:

- **МедАнгел:** ЛИС (Лабораторная информационная система) – это модуль программы МедАнгел для автоматизации медицинской лаборатории и поэтапного контроля учета и обмена данными. Модуль может быть использован как отдельный программный продукт, так и в составе МИС МедАнгел.

- **Altey Laboratory** – лабораторная информационная система, предназначенная для комплексной автоматизации всех основных видов лабораторных исследований: биохимических, гематологических, клинических, коагулологических, серологических, иммунологических, аллергологических и прочих.

- **1С:Медицина** – это программное решение, предназначенное для автоматизации процессов в медицинских учреждениях. Его функциональные возможности включают:

1. Автоматизация взятия проб и регистрация входящего материала с контролем объема и отбраковкой непригодных проб.

2. Штрихкодová маркировка проб с индивидуальными настройками макетов этикеток.
3. Поддержка сортировки проб по штрих-коду и управление их движением.
4. Регистрация различных типов результатов, включая количественные и качественные данные, изображения и текстовые заключения.
5. Взаимодействие с лабораторными анализаторами в различных режимах: по штрих-коду, пакетном и однонаправленном.
6. Возможность ручной регистрации результатов и расчетных показателей.

Эти функции значительно упрощают работу медицинских учреждений, повышая эффективность процессов и качество обслуживания пациентов.

Таким образом, нами был выполнен поиск информации по предметной области, выделены ключевые определения, которые помогут улучшить разрабатываемый программный модуль. На основании этого нами построена диаграмма вариантов использования и ER-диаграмма, а также проведен анализ программ аналогов.

2. РАЗРАБОТКА ТЕХНИЧЕСКОГО ЗАДАНИЯ

Разработка данного технического задания проводилась согласно документу ГОСТ 19.201-78 [1].

Введение

Наименование программы – «Лаборатория».

Программа предназначена для учета анализов медицинской лаборатории. Она будет использоваться в медицинских учреждениях для автоматизации процессов управления данными о пациентах и результатах анализов, что позволит повысить эффективность работы лаборатории и улучшить качество обслуживания пациентов.

Основания для разработки

Разработка программы ведется на основании учебного плана и перечня тем, утвержденных на заседании предметно-цикловой комиссии по информатике и программированию.

Назначение разработки

Программное обеспечение «Лаборатория» предназначено для автоматизации учета анализов медицинской лаборатории, включая функции добавления клиентов, формирования заказов, маркировки образцов, сохранения данных анализов и формирования отчетов.

Требования к программе

Программа должна обеспечивать выполнение следующих функций:

- Добавление новых клиентов в базу данных.

Данная функция позволяет медицинскому регистратору вводить и сохранять информацию о новых пациентах, которые обращаются в медицинскую лабораторию.

- Формирование заказов на оказание услуг.

Эта функция позволяет создавать заказы на проведение анализов для пациентов.

- Маркировка образцов биологического материала.

Функция маркировки образцов критически важна для обеспечения правильной идентификации и отслеживания проб.

- Сохранение данных анализов (результатов исследований) в базу данных.
- Формирование отчетов по результатам анализов.

Программа должна обеспечивать:

- Устойчивое функционирование при нормальных условиях эксплуатации.
- Контроль входной и выходной информации.
- Время восстановления после отказа не более 10 минут.

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

Для работы программы необходимы следующие технические средства:

- Оперативная память не менее 4 ГБ.
- Жесткий диск с объемом не менее 500 ГБ.

Исходные коды программы должны быть реализованы на языке C#. В качестве интегрированной среды разработки программы должна быть использована среда программирования Microsoft Visual Studio 2022.

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

Программное обеспечение поставляется в виде изделия на CD диске.

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

Программа должна обеспечивать взаимодействие с пользователем посредством графического пользовательского интерфейса.

Предварительный состав программной документации включает:

- Техническое задание.
- Руководство оператора.

Этапы разработки:

1. Анализ требований и проектирование.
2. Разработка программного кода.
3. Тестирование и отладка.
4. Внедрение и сопровождение: по мере необходимости.

Программное обеспечение должно демонстрировать высокую экономическую эффективность за счет сокращения времени обработки данных и уменьшения вероятности ошибок в учете анализов. Ожидается, что внедрение системы приведет к повышению производительности работы лаборатории и улучшению качества обслуживания пациентов.

Разработанное техническое задание на создание программного модуля для автоматизации учета анализов медицинской лаборатории содержит подробное описание функциональных и технических требований к разрабатываемой системе.

3. ОПИСАНИЕ АЛГОРИТМОВ И ФУНКЦИОНИРОВАНИЯ ПРОГРАММЫ

Алгоритм выполнения программы для роли «Медсестра» изображен на рисунке 3, в нем отражаются функции, доступные данной роли, в упрощенном виде.

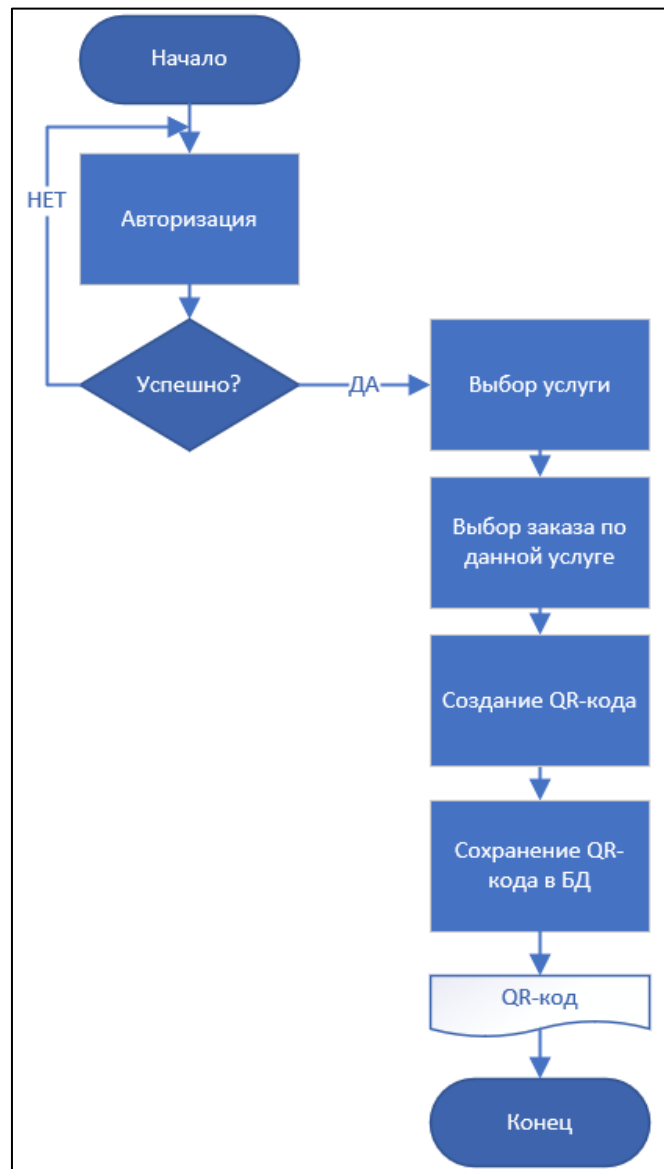


Рисунок 3 – Алгоритм программы для медсестры

Для разрабатываемого программного модуля была создана база данных в SQL Server Management Studio (SSMS), которая хранит информацию о пациентах, сотрудниках и результатах исследований. Эта база данных

организована с использованием реляционной модели, что позволяет эффективно структурировать данные в виде таблиц, каждая из которых содержит строки и столбцы, представляющие отдельные записи и их атрибуты.

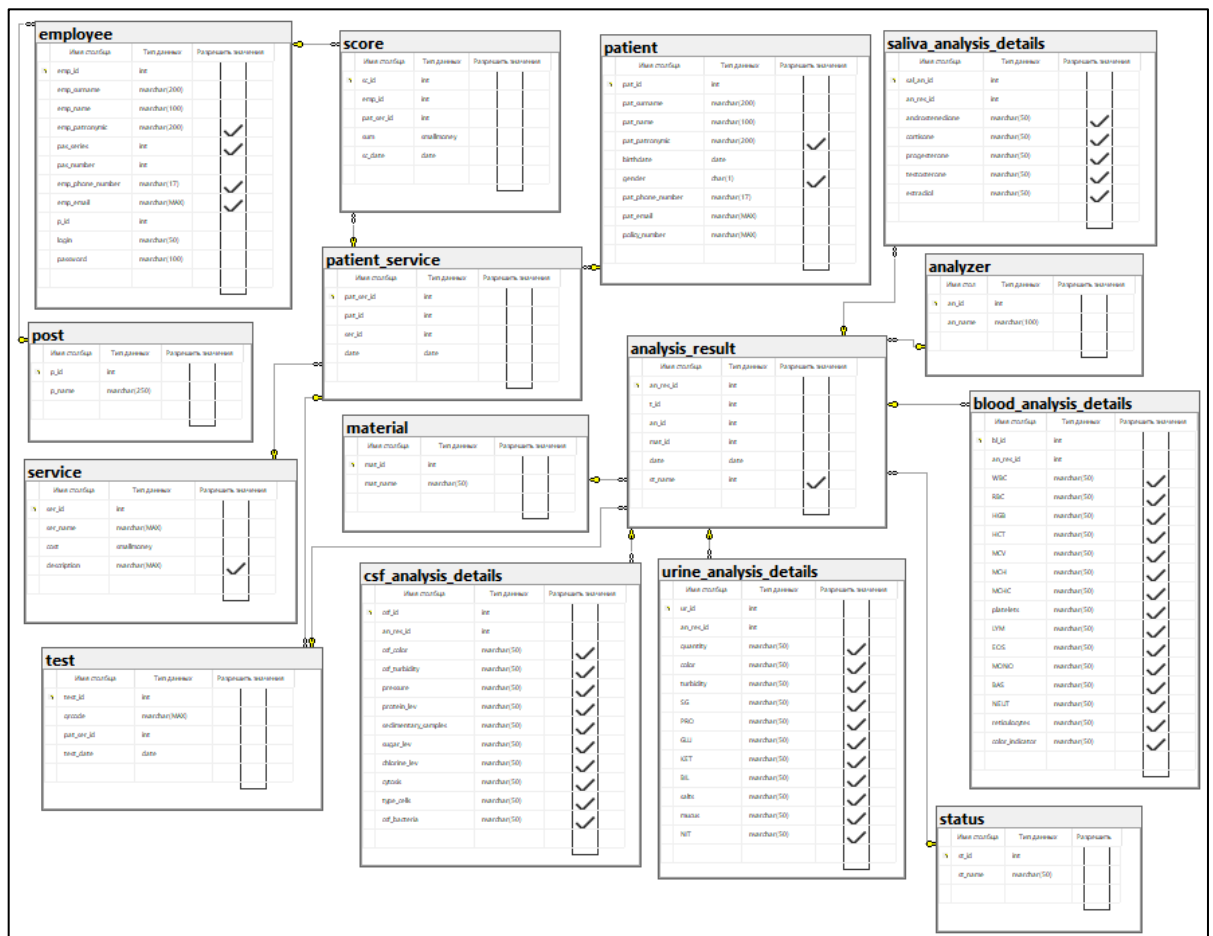
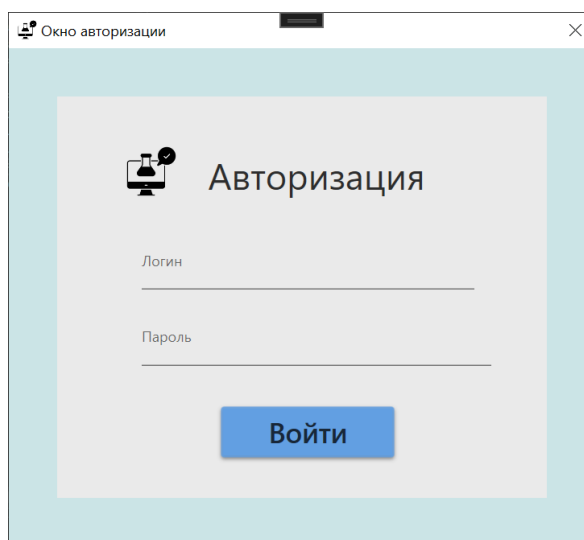


Рисунок 4 – Диаграмма базы данных

Сам программный модуль для учета анализов медицинской лаборатории был создан с использованием среды Visual Studio, языка программирования C# и платформы Windows Presentation Foundation (WPF) [Приложение 1]. Этот модуль обеспечивает эффективное управление данными и автоматизацию процессов в лаборатории, позволяя улучшить взаимодействие между пользователями и системой. WPF предоставляет возможности для создания интуитивно понятного интерфейса, что облегчает работу с программой и повышает производительность сотрудников.

При запуске программы пользователю отображается форма авторизации, на которой он должен ввести логин и пароль.

В зависимости от роли пользователя (регистратор, медсестра или лаборант) при входе в систему будет открываться соответствующее окно с уникальным набором функций.



Окно авторизации

Авторизация

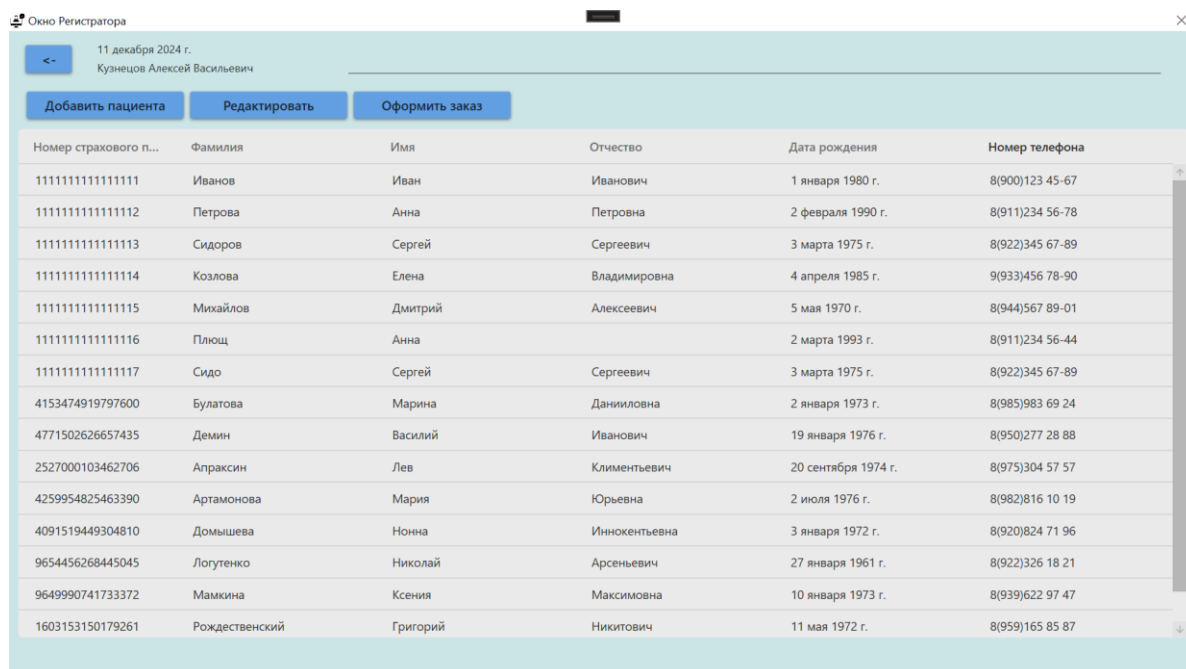
Логин

Пароль

Войти

Рисунок 5 – Форма авторизации

Регистратор получит доступ к окну, где сможет просматривать список пациентов лаборатории.



Окно Регистратора

11 декабря 2024 г.
Кузнецов Алексей Васильевич

Добавить пациента Редактировать Оформить заказ

Номер страхового п...	Фамилия	Имя	Отчество	Дата рождения	Номер телефона
111111111111111	Иванов	Иван	Иванович	1 января 1980 г.	8(900)123 45-67
111111111111112	Петрова	Анна	Петровна	2 февраля 1990 г.	8(911)234 56-78
111111111111113	Сидоров	Сергей	Сергеевич	3 марта 1975 г.	8(922)345 67-89
111111111111114	Козлова	Елена	Владимировна	4 апреля 1985 г.	9(933)456 78-90
111111111111115	Михайлов	Дмитрий	Алексеевич	5 мая 1970 г.	8(944)567 89-01
111111111111116	Плющ	Анна		2 марта 1993 г.	8(911)234 56-44
111111111111117	Сидо	Сергей	Сергеевич	3 марта 1975 г.	8(922)345 67-89
4153474919797600	Булатова	Марина	Даниловна	2 января 1973 г.	8(985)983 69 24
4771502626657435	Демин	Василий	Иванович	19 января 1976 г.	8(950)277 28 88
2527000103462706	Апраксин	Лев	Климентьевич	20 сентября 1974 г.	8(975)304 57 57
4259954825463390	Артамонова	Мария	Юрьевна	2 июля 1976 г.	8(982)816 10 19
4091519449304810	Домышева	Нонна	Иннокентьевна	3 января 1972 г.	8(920)824 71 96
9654456268445045	Логутенко	Николай	Арсеньевич	27 января 1961 г.	8(922)326 18 21
9649990741733372	Мамкина	Ксения	Максимовна	10 января 1973 г.	8(939)622 97 47
1603153150179261	Рождественский	Григорий	Никитович	11 мая 1972 г.	8(959)165 85 87

Рисунок 6 – Окно регистратора

При нажатии на кнопку «Добавить пациента», регистратор сможет ввести информацию о пациенте и сохранить его в базу данных.

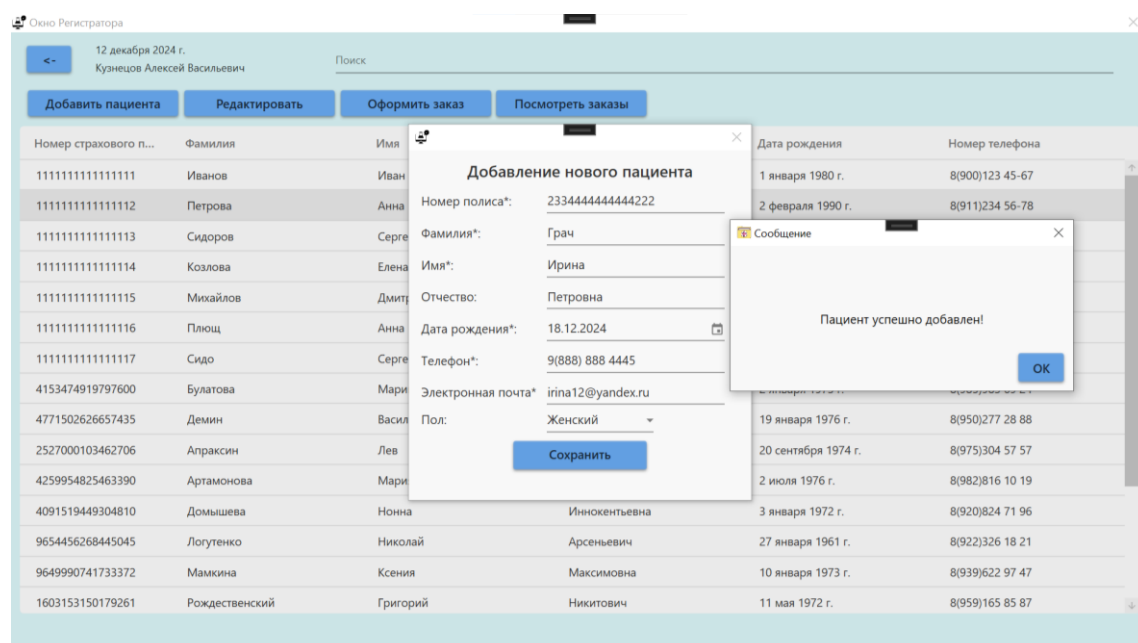


Рисунок 7 – Добавление нового пациента в БД

А при выборе пациента и нажатии на кнопку «Редактировать», откроется окно редактирования информации о пациенте с кнопкой для сохранения изменений в базу данных.

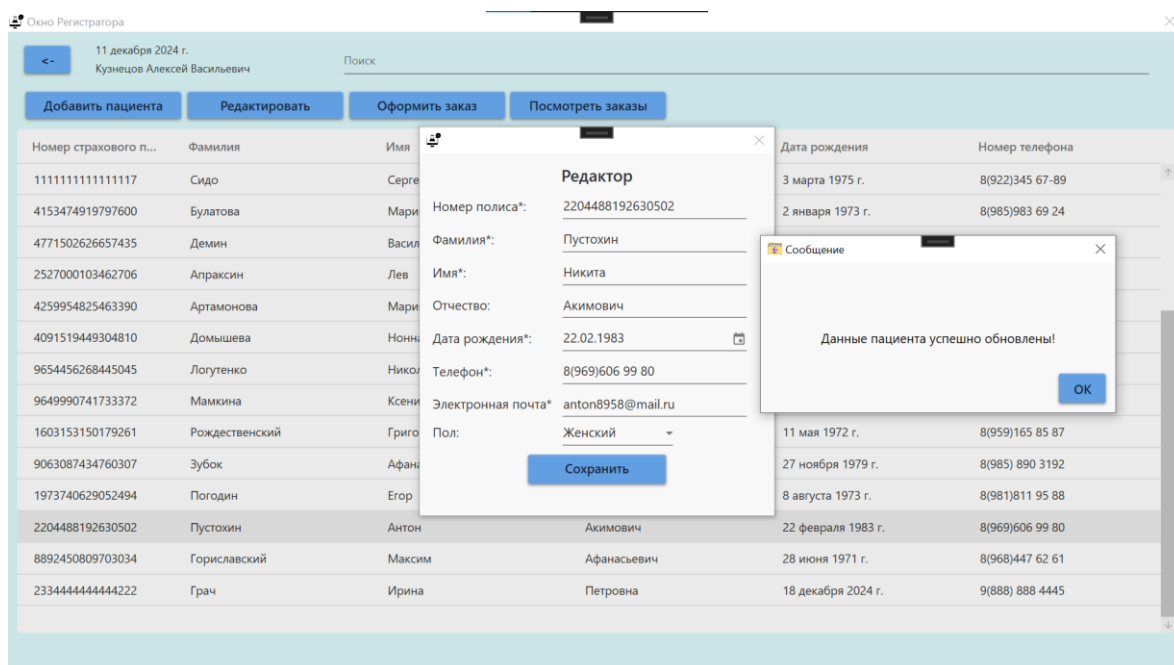


Рисунок 8 – Редактирование информации о пациенте

На рисунке 9 показано окно формирования заказа на сдачу анализов для выбранного пациента.

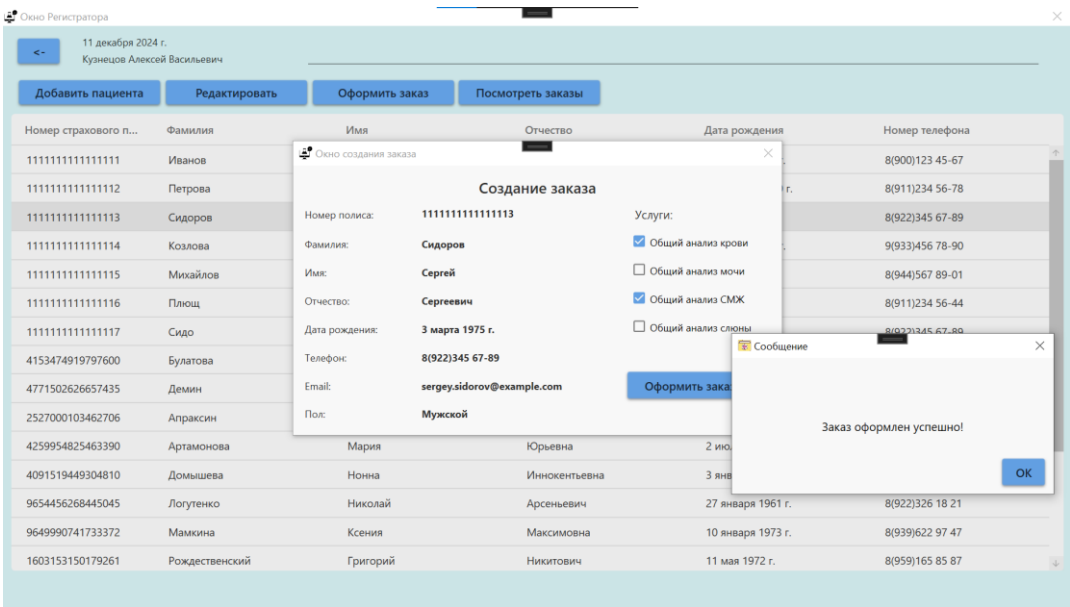


Рисунок 9 – Окно формирования заказа

Для просмотра заказов, которые оформил пациента, нужно выбрать этого пациента из списка и нажать на кнопку «Посмотреть заказы». Список заказов отобразится в отдельном окне.

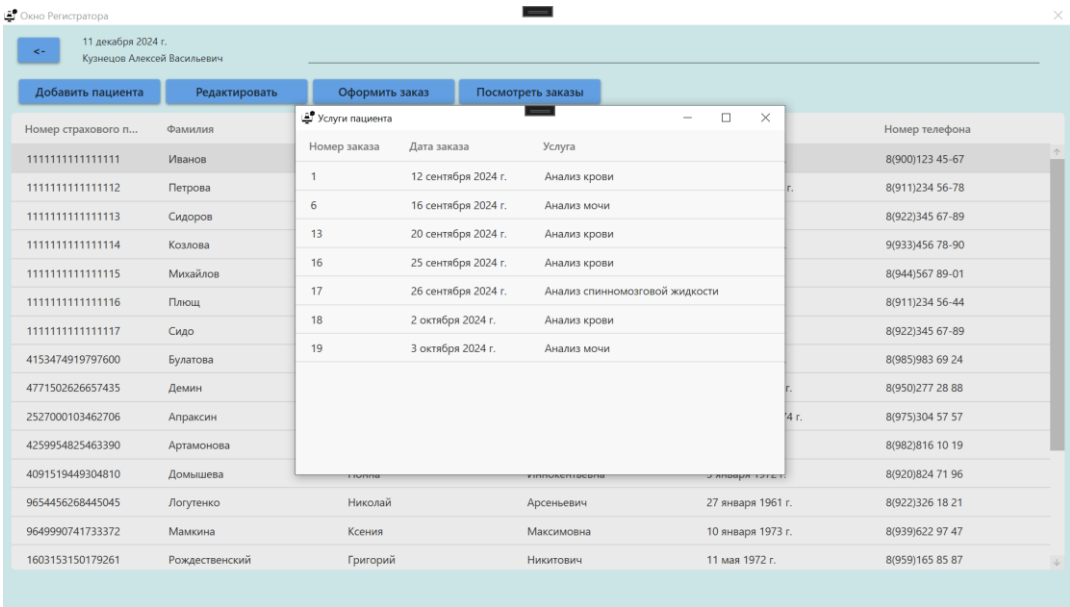


Рисунок 10 – Окно заказов

Медсестра может при выборе нужной услуги и заказа сформировать QR-код для пробирки. Он формируется автоматически и содержит информация – ФИО пациента, номер страхового полиса пациента, наименование услуги и дата

взятия биоматериала. После формирования QR-кода выйдет окно с ним (Рисунок 11), его можно считать с экрана ПК, или сохранить в файл формата .pdf (Рисунок 12).

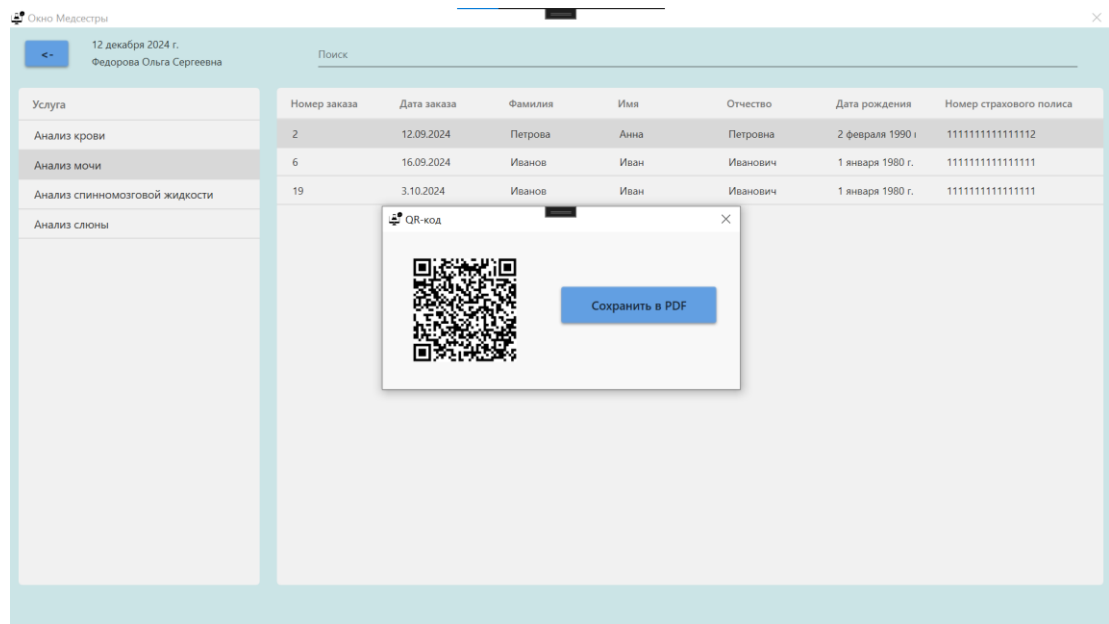


Рисунок 11 – QR-код

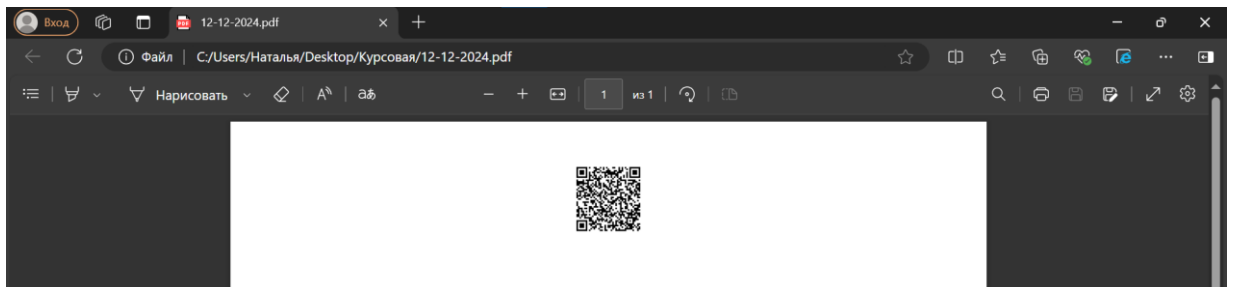


Рисунок 12 – Сохранение QR-кода в PDF файл

Лаборант, в свою очередь, получит доступ к окну для работы с лабораторными данными (Рисунок 13), где сможет вводить результаты анализов (Рисунок 14) и генерировать отчеты по этим данным в документе.

На рисунке 15 сформирован отчет по анализу крови.

Такой функционал обеспечивает удобство и эффективность работы, позволяя лаборанту сосредоточиться на своих основных задачах.

Таким образом, каждый пользователь будет иметь доступ только к тем функциям, которые соответствуют его роли, что повышает безопасность и

удобство работы с системой. Такой подход позволяет свести к минимуму риски несанкционированного доступа к конфиденциальной информации и гарантирует, что пользователи смогут эффективно выполнять свои задачи, не отвлекаясь на ненужные функции.

12 декабря 2024 г.
Иванова Мария Петровна

Поиск

Номер пробы	Дата взятия...	Услуга
1	12.09.2024	Анализ крови
2	12.09.2024	Анализ мочи
4	12.09.2024	Анализ спинномозговой жидк.
5	5.12.2024	Анализ слюны
6	7.12.2024	Анализ спинномозговой жидк.
7	7.12.2024	Анализ слюны
8	7.12.2024	Анализ спинномозговой жидк.

Параметр анализа	Значение
Лейкоциты	112
Эритроциты	12
Гемоглобин	100
Гематокрит	0.3
Средний объем эритроцита	60
Среднее содержание гемоглобина в эритроците	20
Средняя концентрация гемоглобина в эритроците	39
Тромбоциты	12
Лимфоциты	12
Эозинофилы	23
Моноциты	34
Базофилы	35
Нейтрофилы	46
Ретикулоциты	57
Цветовой показатель	66

Печать результатов

Рисунок 13 – Окно лаборанта

Преимущества многоуровневой модели доступа:

Безопасность: Ограничение доступа к информации снижает вероятность утечек данных.

Эффективность: Пользователи могут сосредоточиться на выполнении своих задач без лишних отвлекающих факторов.

Удобство: интерфейс адаптирован под конкретные роли, что упрощает работу с системой.

Таким образом, внедрение такой модели управления данными в лаборатории не только повышает качество работы, но и способствует более быстрому и точному выполнению анализов и составлению отчётов.

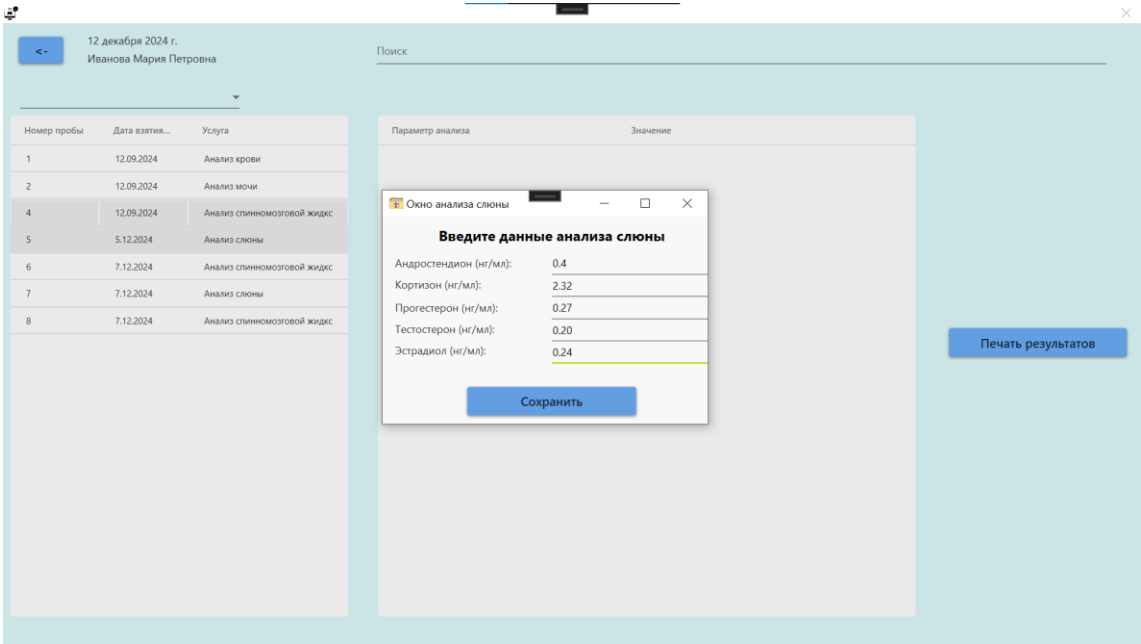


Рисунок 14 – Ввод результатов анализов

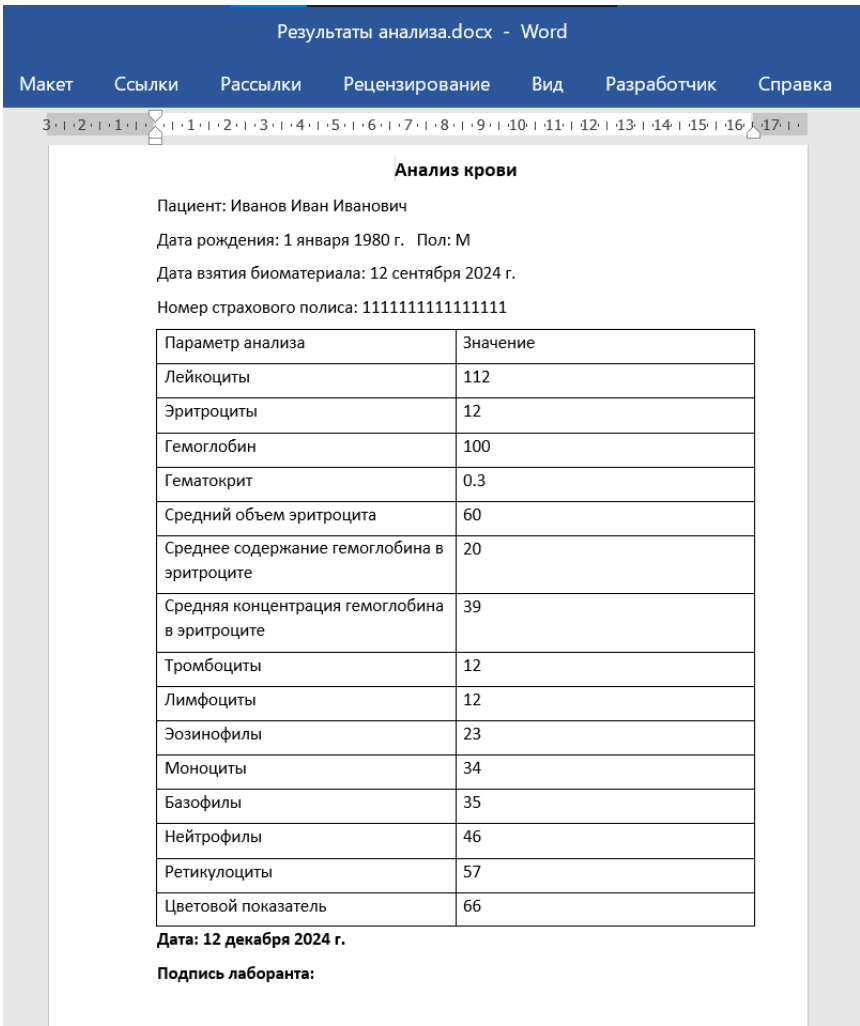


Рисунок 15 – Отчет по анализу крови

В дополнение к этому, программа включает метод, который используется для проверки заполнения обязательных полей при добавлении нового пациента (Рисунок 16). Этот механизм не только обеспечивает целостность данных, но и предотвращает ошибки, которые могут возникнуть в результате неполного или некорректного ввода информации. Например, метод `ValidateInputs` проверяет, заполнены ли все обязательные поля, такие как номер полиса, фамилия, имя, дата рождения, телефон и электронная почта.

```
private bool ValidateInputs()
{
    return !string.IsNullOrEmpty(policyNumberTB.Text) &&
        !string.IsNullOrEmpty(surnameTB.Text) &&
        !string.IsNullOrEmpty(nameTB.Text) &&
        birthdateDP.SelectedDate != null &&
        !string.IsNullOrEmpty(phoneTB.Text) &&
        IsValidEmail(emailTB.Text);
}
```

Рисунок 16 – Метод проверки обязательных полей

Кроме того, для улучшения пользовательского опыта в программе реализован метод, позволяющий использовать маску для поля «Номер телефона» (Рисунок 17). Это помогает пользователям вводить номер телефона в стандартизированном формате, что также снижает вероятность ошибок при вводе.

Для стилизации программы были использованы пакеты NuGet `MaterialDesignColors` и `MaterialDesignThemes`, что придает приложению современный и привлекательный вид. Также для расширения функциональности и взаимодействия с операционной системой в программе были задействованы библиотеки `System.IO` и `Microsoft.Win32`.

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

```

private void phoneTB_PreviewTextInput(object sender, TextCompositionEventArgs e)
{
    if (!char.IsDigit(e.Text, 0))
    {
        e.Handled = true;
        return;
    }
    string currentText = phoneTB.Text.Replace("(", "").Replace(")", "").Replace("-", "").Replace(" ", "");
    if (currentText.Length >= 12)
    {
        e.Handled = true;
        return;
    }
    if (currentText.Length == 0)
        phoneTB.Text += e.Text;
    else if (currentText.Length == 1)
        phoneTB.Text += "(" + e.Text;
    else if (currentText.Length < 4)
        phoneTB.Text += e.Text;
    else if (currentText.Length == 4)
        phoneTB.Text += ") " + e.Text;
    else if (currentText.Length < 7)
        phoneTB.Text += e.Text;
    else if (currentText.Length == 7)
        phoneTB.Text += " " + e.Text;
    e.Handled = true;
    phoneTB.CaretIndex = phoneTB.Text.Length;
}

```

Рисунок 17 – Метод, реализующий маску для номера телефона

4. ТЕСТИРОВАНИЕ ПРОГРАММНОГО МОДУЛЯ

В рамках тестирования разработанного программного модуля для автоматизации учета анализов медицинской лаборатории были проведены следующие тесты.

Тестирование добавления новых пациентов в базу данных

Цель: Убедиться, что информация о новых пациентах корректно сохраняется в базе данных.

- Сценарий: Добавление нового пациента с корректными данными.

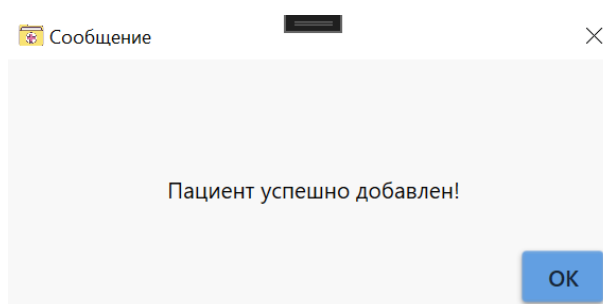


Рисунок 18 – Сообщение об успешном добавлении пациента

Ожидаемый результат: Сообщение об успешном добавлении пациента.

Полученный результат: Сообщение об успешном добавлении пациента.

Решение проблемы: Убедиться, что все поля формы заполнены корректно, и обеспечить валидацию данных.

- Сценарий: Попытка добавить клиента с неполными данными.

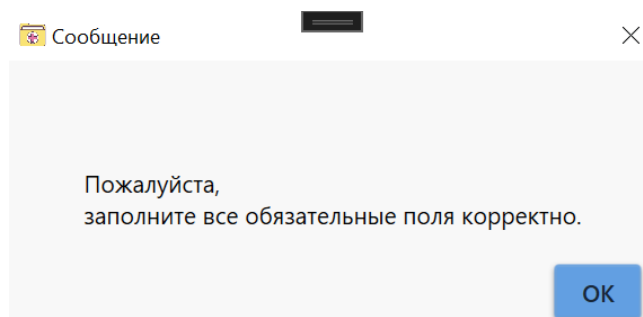


Рисунок 19 – Сообщение об ошибке

Ожидаемый результат: Сообщение об ошибке с указанием на обязательные поля.

Полученный результат: Сообщение об ошибке.

Решение проблемы: Реализовать проверку на заполненность обязательных полей перед добавлением клиента.

Тестирование сохранения данных анализов в базу данных

Цель: Убедиться, что результаты исследований корректно сохраняются.

- Сценарий: Сохранение корректных данных анализа.

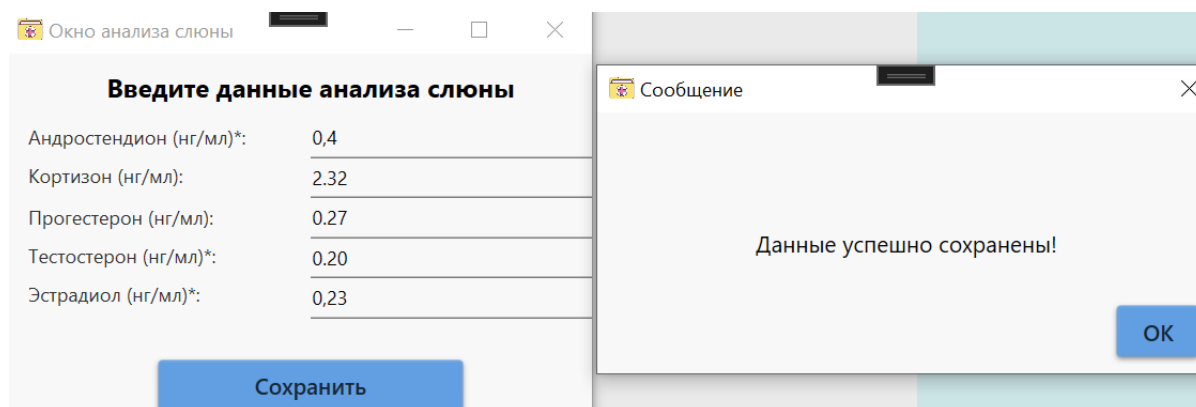


Рисунок 20 – Сообщение об успешном сохранении данных

Ожидаемый результат: Сообщение о успешном сохранении данных.

Полученный результат: Сообщение о успешном сохранении данных.

Решение проблемы: Убедиться, что все поля для данных анализа заполнены корректно.

- Сценарий: Попытка сохранения данных анализа с некорректными данными.

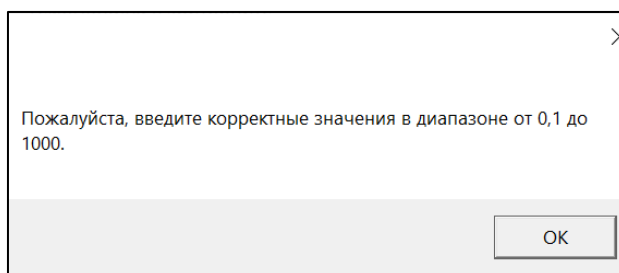


Рисунок 21 – Сообщение об ошибке с указанием на некорректные данные

Ожидаемый результат: Сообщение об ошибке с указанием на некорректные данные.

Полученный результат: Сообщение об ошибке.

Решение проблемы: Реализовать валидацию данных перед сохранением.

Тестирование программы осуществлялось на персональном компьютере со следующими техническими характеристиками:

- Процессор – AMD 3020e with Radeon Graphics 1.20 GHz.
- Оперативная память – 4,00 ГБ.
- Видеокарта – AMD Radeon(TM) Graphics.
- Операционная система – Windows 10 Pro.

Тестирование программного модуля продемонстрировало его надежность и функциональность. Все проведенные тесты подтвердили, что система соответствует заявленным требованиям и обеспечивает удобство работы для пользователей. В дальнейшем планируется продолжение тестирования с целью выявления и устранения возможных недочетов, а также внедрение новых функций для улучшения пользовательского опыта.

5. РУКОВОДСТВО ОПЕРАТОРА

Данное руководство предназначено для лаборантов, использующих программу для учета анализов медицинской лаборатории, которая поставляется на компакт-диске (CD) [Приложение 2]. Программа позволяет отображать результаты анализов, вводить новые данные и распечатывать результаты.

Вставьте CD в дисковод

Поместите компакт-диск с программой в дисковод вашего компьютера.

Запустите установочный файл

Откройте «Мой компьютер» или «Этот компьютер».

В списке устройств найдите CD-диск и откройте его.

Дважды щелкните по файлу «setup.exe» для начала установки.

Следуйте инструкциям установщика

Следуйте указаниям на экране для завершения установки программы на ваш компьютер.

Развертывание базы данных:

Перед использованием программы необходимо развернуть скрипт базы данных в SQL Server.

- Откройте SQL Server Management Studio.
- Подключитесь к вашему серверу баз данных.
- Создайте новую базу данных или выберите существующую.
- Скопируйте и выполните SQL-скрипт, предоставленный на CD вместе с программным обеспечением.

Настройка конфигурационного файла

После развертывания базы данных настройте подключение в конфигурационном файле программы.

Убедитесь, что указаны правильные параметры подключения (сервер, имя базы данных, учетные данные).

Запустите программу

После завершения установки, дважды щелкните по иконке программы «Лаборатория.exe» на рабочем столе или в папке, куда была установлена программа.

Вход в систему

Введите логин и пароль в окне авторизации (Рисунок 22).

Нажмите кнопку «Войти».

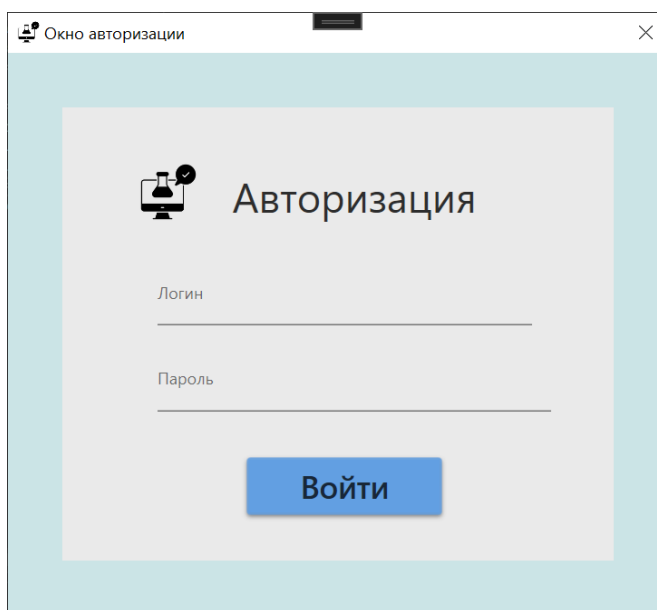


Рисунок 22 – Окно авторизации

Основные элементы интерфейса

После успешной авторизации вы увидите главное окно лаборанта (Рисунок 23), которое представляет собой интуитивно понятный и удобный интерфейс для работы с лабораторными данными.

Структура интерфейса:

- Список доступных проб.

Слева на экране отображается список доступных проб, который позволяет нам быстро ориентироваться в текущих образцах.

- Результаты анализа.

Справа от списка проб отображаются результаты анализа, что позволит вам сразу видеть введенные данные.

- Выпадающий список с названиями услуг.

На форме имеется выпадающий список с названиями услуг, который предоставляет вам возможность выбора необходимых анализов или процедур. Это делает процесс работы более гибким и позволяет вам быстро переключаться между различными услугами в зависимости от требований текущей задачи.

- Элемент поиска.

В правом верхнем углу интерфейса находится элемент поиска, который значительно упрощает поиск нужной информации. С его помощью вы сможете быстро находить конкретные пробы или результаты анализов, что экономит время и повышает общую эффективность работы.

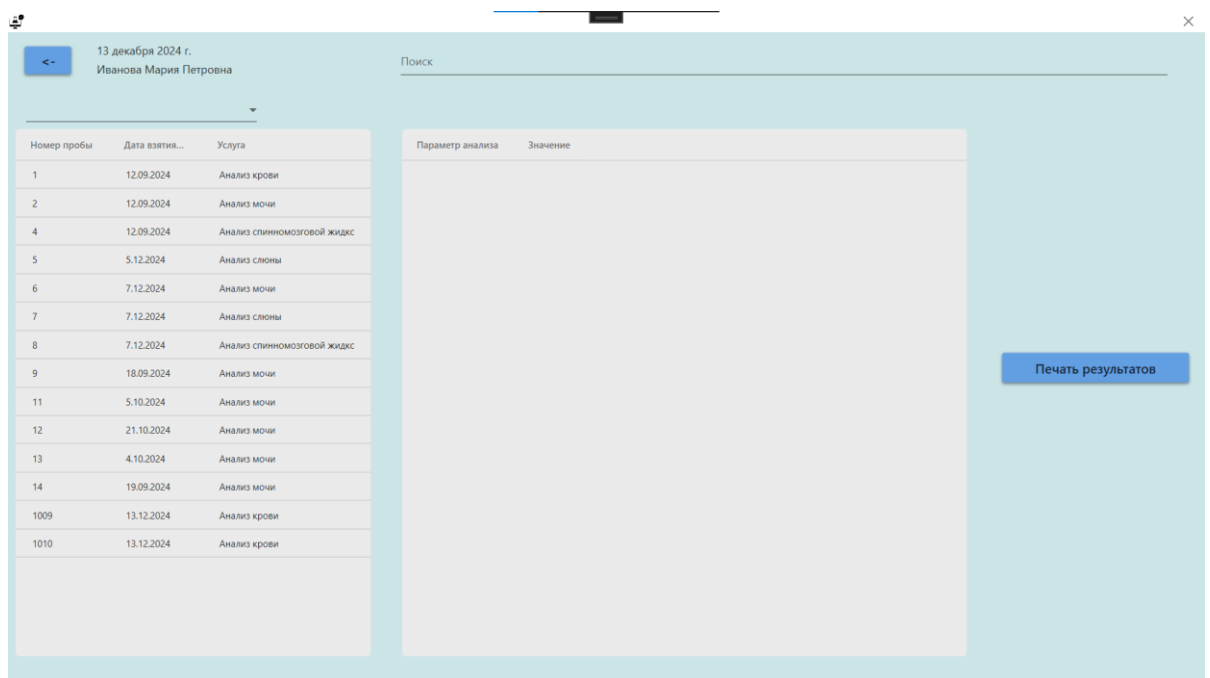


Рисунок 23 – Главное окно лаборанта

Выбор записи

В левом списке выберите интересующий вас анализ.

Программа автоматически проверит наличие данных для выбранного анализа.

Если данные доступны

Результаты анализа будут отображены в правой части окна.

Вы можете просмотреть все параметры анализа и их значения (Рисунок 24).

Номер пробы	Дата взятия...	Услуга
1	12.09.2024	Анализ крови
2	12.09.2024	Анализ мочи
4	12.09.2024	Анализ спинномозговой жидк.
5	5.12.2024	Анализ слюны
6	7.12.2024	Анализ спинномозговой жидк.
7	7.12.2024	Анализ слюны
8	7.12.2024	Анализ спинномозговой жидк.

Параметр анализа	Значение
Лейкоциты	112
Эритроциты	12
Гемоглобин	100
Гематокрит	0.3
Средний объем эритроцита	60
Среднее содержание гемоглобина в эритроците	20
Средняя концентрация гемоглобина в эритроците	39
Тромбоциты	12
Лимфоциты	12
Эозинофилы	23
Моноциты	34
Базофилы	35
Нейтрофилы	46
Ретикулоциты	57
Цветовой показатель	66

Печать результатов

Рисунок 24 – Результаты анализа

Если данные отсутствуют

Появится окно ввода данных, в котором вам будет предложено ввести результаты анализа.

Введите необходимые данные в соответствующие поля и нажмите кнопку «Сохранить» (Рисунок 25).

Введите данные анализа мочи

Количество (мл):

Кетоновые тела (ммоль/л): ☐ отрицательно

Цвет: ☐ отрицательно

Билирубин: ☐ отрицательно

Соли: ☐ отрицательно

Мутность:

Слизь: ☐ не обнаружена

Удельный вес:

Белок (г/л): ☐ не обнаружены

Глюкоза (ммоль/л): ☐ отрицательно

Бактерии: ☐ не обнаружены

Сохранить

Рисунок 25 – Окно для ввода результатов

Нажмите кнопку «Печать результатов» на главном окне (Рисунок 26).



Теперь вы знакомы с основными функциями программы для учета анализов медицинской лаборатории, установленной с компакт-диска. Если у вас возникнут вопросы или проблемы, обратитесь к вашему администратору или технической поддержке.

ЗАКЛЮЧЕНИЕ

В ходе выполнения курсового проекта была разработана программная система для автоматизации учета анализов медицинской лаборатории. Данная система направлена на повышение эффективности работы лабораторий, оптимизацию процессов учета, обработки и хранения данных о результатах исследований.

Созданный программный модуль обладает интуитивно понятным интерфейсом и логичной структурой, что делает его простым и удобным в использовании для лаборантов. Автоматизация рутинных операций, таких как ввод данных о результатах анализов и формирование отчетов, значительно экономит время сотрудников и позволяет им сосредоточиться на более важных задачах.

Программный модуль может быть успешно внедрен в медицинские лаборатории различного масштаба и типа, что позволит существенно оптимизировать процессы управления и повысить общую эффективность работы. В дальнейшем планируется расширить функциональность модуля, добавив возможности управления выдачей и возвратом образцов, а также интеграцию с другими информационными системами и базами данных.

Таким образом, реализация данного курсового проекта создает основу для создания полноценной автоматизированной системы управления медицинской лабораторией, соответствующей современным требованиям и ожиданиям пользователей. Система будет способствовать улучшению качества диагностики и повышению уровня медицинского обслуживания, что является важной задачей в условиях современного здравоохранения.

СПИСОК ЛИТЕРАТУРЫ

1. ГОСТ 19.201-78. Единая система программной документации. Техническое задание. Требования к содержанию и оформлению: национальный стандарт Российской Федерации: дата введения – 1 января 1980 г. / Федеральное агентство по техническому регулированию и метрологии. – Изд. официальное. – Москва: Стандартинформ, 2010.

2. ГОСТ Р 53079.4-2008. Технологии лабораторные клинические. Обеспечение качества клинических лабораторных исследований. Часть 4. Правила ведения преаналитического этапа: национальный стандарт Российской Федерации: дата введения - 2009 г. / Федеральное агентство по техническому регулированию и метрологии. – Изд. официальное. – Москва: Стандартинформ, 2009. – 2-5 с.

3. ГОСТ Р ИСО 15189-2015. Лаборатории медицинские. Частные требования к качеству и компетентности: национальный стандарт Российской Федерации: дата введения - 1 июня 2016 г. / Федеральное агентство по техническому регулированию и метрологии. – Изд. официальное. – Москва: Стандартинформ, 2015 3 – 3с.

4. Приказ Министерства труда и социальной защиты Российской Федерации от 14 марта 2018 года N 145н «Профессиональный стандарт «Специалист в области клинической лабораторной диагностики»» [Электронный ресурс]. – Режим доступа: <https://classinform.ru/profstandarty/02.032-spetsialist-v-oblasti-clinicheskoi-laboratornoi-diagnostiki.html>

5. Дядя Г. И. и др. Универсальный справочник практикующего врача (Раздел «Общий анализ крови»). – Воронеж: Научная книга, 2017. – 512 с. ISBN 978-5-521-05469-5.

6. Кишкун А. А. Клиническая лабораторная диагностика. Учебное пособие (Глава 2. Гематологические исследования). – М.: ГЭОТАР-Медиа, 2015. – 976 с. ISBN 978-5-9704-3518-2.

7. Кузнецов, А. И. Основы автоматизации лабораторных процессов. – СПб.: Питер, 2019. – 256 с.
8. Лабораторная диагностика: все лабораторные исследования для диагностики и лечения : [пер. с англ.] / Уоллах Ж. ; отв. ред. О. Шестова. - 8-е изд. - М. : Эксмо, 2013 - 1358, [1] с. : ил. - (Медицинская энциклопедия). - Текст: непосредственный.
9. Лебедев, С. А. Современные подходы к автоматизации медицинских лабораторий. – М.: ГЭОТАР-Медиа, 2021. – 400 с.
10. Медицинские лабораторные технологии руководство по клинической лабораторной диагностике, в 2 т. / [проф. В.В. Алексеев и др.]; под ред. проф. А.И. Карпищенко. — 3-е изд.. — Москва : ГЭОТАР-Медиа, 2012. —; 30. — ISBN 978-5-9704-2276-2.
11. Методы клинических лабораторных исследований [Текст] : [учебник] / Камышников В. С., Волотовская О. А., Ходюкова А. Б. и др. ; под ред. В. С. Камышникова . - 7-е изд. . - М. : МЕДпресс-информ , 2015 . - 735, с.: ил., цв. ил.
12. Михайлова, Е. В. Информационные технологии в здравоохранении: учебное пособие. – М.: Флинта, 2020. – 320 с.
13. Назаренко Г. И., Кишкун А. А. Клиническая оценка результатов лабораторных исследований. – М.: Медицина, 2006. – 544 с. ISBN 5-225-04579-0.
14. Ольховик А. Ю., Садовников П. С., Васильев А. В., Денисов Д. Г. Сравнительная оценка показателей общеклинического исследования венозной и капиллярной крови // Medline.ru. — 11.06.2017. — Т. 18. — С. 113-122.
15. Петров, С. В. Лабораторная диагностика: современные методы и технологии. – СПб.: Питер, 2021. – 320 с. ISBN 978-5-4461-0123-4.

ПРИЛОЖЕНИЯ

КОД ПРОГРАММА

Файл MainWindow.xaml

```

<Window x:Class="WpfApp.MainWindow"
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
xmlns:local="clr-namespace:WpfApp"
xmlns:materialDesign="http://materialdesigninxaml.net/winfx/xaml/themes"
mc:Ignorable="d" Title="Окно авторизации"
Height="450" Width="500" ResizeMode="NoResize"
WindowStartupLocation="CenterScreen"
Icon="/Image/Lab.ico">
    <Grid Background="#FFCBE4E6">
        <Label Background="#EAEAEA" Margin="40"/>
        <Label Content="Авторизация" FontSize="30"
Margin="161,81,118,0" VerticalAlignment="Top"/>
        <Image Source="/Image/Lab.png"
Margin="97,81,0,0" Height="42"
VerticalAlignment="Top"
HorizontalAlignment="Left" Width="49"/>
        <TextBox x:Name="loginTB"
Margin="110,152,100,0"
materialDesign:HintAssist.Hint="Логин" Height="48"
VerticalAlignment="Top" CaretBrush="#FF629FE2"
SelectionBrush="#FF629FE2"/>
        <TextBox x:Name="passwordTB"
Margin="110,215,0,0"
materialDesign:HintAssist.Hint="Пароль"
HorizontalAlignment="Left" Width="289"
Height="48" VerticalAlignment="Top"
CaretBrush="#FF629FE2"
SelectionBrush="#FF629FE2"/>

```

```

<Button x:Name="InputBt" Content="Войти"
FontSize="23" Margin="176,297,166,0"
Click="InputBt_Click" Height="42"
VerticalAlignment="Top" Background="#FF629FE2"
BorderBrush="#FF629FE2"/>
</Grid>
</Window>

```

Файл MainWindow.xaml.cs

```

using System;
using System.Linq;
using System.Windows;
using WpfApp.Model;
namespace WpfApp
{
    /// <summary>
    /// Логика взаимодействия для MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
        }
        private void InputBt_Click(object sender,
RoutedEventArgs e)
        {
            try
            {
                using (var db = new MedLabEntities())
                {
                    var user = db.employee.FirstOrDefault(u
=> u.login == loginTB.Text && u.password ==
passwordTB.Text);
                    if (user != null)
                    {
                        string fullName =
$" {user.emp_surname} {user.emp_name}
{user.emp_patronymic}";

```

```
switch (user.p_id)
{
    case 1:
        RegistryEmployee registry = new
RegistryEmployee();
        registry.Show();
        registry.SurnameLb.Content =
fullName;
        this.Close();
        break;
    case 2:
        MedicalNurse medical = new
MedicalNurse();
        medical.Show();
        medical.surnameLb.Content =
fullName;
        this.Close();
        break;
    case 3:
        LaboratoryAssistant laboratory =
new LaboratoryAssistant();
        laboratory.Show();
        laboratory.surnameLb.Content =
fullName;
        this.Close();
        break;
}
}
else
{
    var customMessageBox = new
CustomMessageBox("Неверный логин или
пароль!");
    customMessageBox.ShowDialog();
}
}
}
catch
{
    var customMessageBox = new
CustomMessageBox("Ошибка при авторизации!");
    customMessageBox.ShowDialog();
```

```

    }
}
}
}

Файл RegistryEmployee.xaml

using System;
using System.Linq;
using System.Windows;
using System.Windows.Controls;
using WpfApp.Model;

namespace WpfApp
{
    /// <summary>
    /// Логика взаимодействия для
RegistryEmployee.xaml
    /// </summary>
    public partial class RegistryEmployee : Window
    {
        MedLabEntities db = new MedLabEntities();
        public RegistryEmployee()
        {
            InitializeComponent();
            DateLb.Content =
DateTime.Today.ToString("D");
        }
        private void Window_Loaded(object sender,
RoutedEventArgs e)
        {
            Load();
            backBtn.Content = "<-";
        }
        private void Load()
        {
            try
            {
                var query = db.patient.ToList();
                PatientDG.ItemsSource = query;
            }
            catch
            {

```

```

        var customMessageBox = new
CustomMessageBox("Ошибка загрузки данных!");
        customMessageBox.ShowDialog();
    }
}

private void searhTB_TextChanged(object sender,
TextChangedEventArgs e)
{
    try
    {
        if (SearhTB.Text != null && SearhTB.Text
!= "")
        {
            string searhText = SearhTB.Text;
            var queryPet = from Patient in db.patient
                where
Patient.pat_surname.Contains(searhText.ToLower())
                ||
Patient.pat_name.Contains(searhText.ToLower())
                ||
Patient.pat_patronymic.Contains(searhText.ToLower()
)
                ||
Patient.policy_number.ToString().Contains(searhText)
                ||
Patient.birthdate.ToString().Contains(searhText)
                select new
            {
                Patient.pat_surname,
                Patient.pat_name,
                Patient.pat_patronymic,
                Patient.policy_number,
                Patient.birthdate,
            };

            PatientDG.ItemsSource =
queryPet.ToList();
        }
        else
        {
            PatientDG.ItemsSource = null;
            Load();
        }
    }
}

```

```

    }
    catch
    {
        var customMessageBox = new
CustomMessageBox("Ошибка при поиске!");
        customMessageBox.ShowDialog();
    }
}

private void RegisterPatientButton_Click(object
sender, RoutedEventArgs e)
{
    //Открытие формы добавления пациента
    PatientRegWindow registrationForm = new
PatientRegWindow();
    registrationForm.ShowDialog();
    Load();
}

private void EditPatientButton_Click(object
sender, RoutedEventArgs e)
{
    if (PatientDG.SelectedItem != null)
    {
        try
        {
            var selectedPatient =
PatientDG.SelectedItem as patient;
            if (selectedPatient != null)
            {
                //Открытие формы редактирования
пациента
                PatientRegWindow editForm = new
PatientRegWindow(selectedPatient);
                editForm.ShowDialog();
                Load();
            }
        }
        catch
        {
            var customMessageBox = new
CustomMessageBox("Ошибка редактирования
пациента!");
            customMessageBox.ShowDialog();
        }
    }
}

```

```

    }
}
else
{
    var customMessageBox = new
CustomMessageBox("Выберите пациента для
редактирования!");
    customMessageBox.ShowDialog();
}
}

private void backBtn_Click(object sender,
RoutedEventArgs e)
{
    MainWindow mainWindow = new
MainWindow();
    mainWindow.Show();
    this.Close();
}

private void CreateOrderTB_Click(object sender,
RoutedEventArgs e)
{
    if (PatientDG.SelectedItem != null)
    {
        var selectedPatient = PatientDG.SelectedItem
as patient;
        if (selectedPatient != null)
        {
            //Открытие формы создания заказа
            PatientServiceWindow editForm = new
PatientServiceWindow(selectedPatient);
            editForm.ShowDialog();
        }
    }
    else
    {
        var customMessageBox = new
CustomMessageBox("Выберите пациента для
создания заказа!");
        customMessageBox.ShowDialog();
    }
}

```

```

private void PatientDG_SelectionChanged(object
sender, SelectionChangedEventArgs e)
{
    SeeOrderTB.Visibility = Visibility.Visible;
}

private void SeeOrderTB_Click(object sender,
RoutedEventArgs e)
{
    if (PatientDG.SelectedItem != null)
    {
        var selectedPatient = PatientDG.SelectedItem
as patient;
        if (selectedPatient != null)
        {
            var queryPetOrders = from Patient in
db.patient
                                join ser in db.patient_service
on Patient.pat_id equals ser.pat_id
                                join se in db.service on
ser.ser_id equals se.ser_id
                                where Patient.pat_id ==
selectedPatient.pat_id
                                select ser;
            if (queryPetOrders.Any())
            {
                //Открытие формы заказов
                PatientOrdersWindow editForm = new
PatientOrdersWindow(selectedPatient);
                editForm.ShowDialog();
            }
            else
            {
                var customMessageBox = new
CustomMessageBox("Заказов нет!");
                customMessageBox.ShowDialog();
            }
        }
    }
    else
    {

```

```

var customMessageBox = new
CustomMessageBox("Выберите пациента для
редактирования!");
customMessageBox.ShowDialog();
    }
}
}
}

```

Файл RegistryEmployee.xaml

```

<Window x:Class="WpfApp.RegistryEmployee"
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
xmlns:materialDesign="http://materialdesigninxaml.net/winfx/xaml/themes"
xmlns:local="clr-namespace:WpfApp"
mc:Ignorable="d" Title="Окно Регистратора"
Loaded="Window_Loaded" Height="550"
Width="1000" ResizeMode="NoResize"
WindowState="Maximized" Icon="/Image/Lab.ico">
<Grid Background="#FFCBE4E6">
<Grid.RowDefinitions>
<RowDefinition Height="60"/>
<RowDefinition/>
</Grid.RowDefinitions>
<Button Background="#FF629FE2"
x:Name="backBtn" ToolTip="Вернуться на форму
авторизации" BorderBrush="#FF629FE2"
Click="backBtn_Click" HorizontalAlignment="Left"
Margin="18,0,0,0" VerticalAlignment="Center"
Height="30" Width="50" FontWeight="Bold"/>
<TextBox x:Name="SearchTB" Grid.Column="0"
Margin="365,16,40,14"
TextChanged="searchTB_TextChanged"
materialDesign:HintAssist.Hint="Поиск"/>

```

```

<Label x:Name="DateLb" Grid.Column="0"
Content="Сер. дата" HorizontalAlignment="Left"
Margin="91,7,0,0" VerticalAlignment="Top"
Width="250"/>
<Label x:Name="SurnameLb" Grid.Column="0"
Content="ФИО врача" HorizontalAlignment="Left"
Margin="91,27,0,0" VerticalAlignment="Top"
Width="250"/>
<DataGrid x:Name="PatientDG" Language="ru-
RU" ColumnWidth="Auto" Grid.Row="1"
AutoGenerateColumns="False" Margin="10,45,10,45"
IsReadOnly="True" Background="#FFEAEEAE"
SelectionChanged="PatientDG_SelectionChanged">
<DataGrid.Columns>
<DataGridTextColumn Header="Номер
страхового полиса" Binding="{Binding
policy_number}" Width="170"/>
<DataGridTextColumn Header="Фамилия"
Binding="{Binding pat_surname}" Width="*/>
<DataGridTextColumn Header="Имя"
Binding="{Binding pat_name}" Width="*/>
<DataGridTextColumn Header="Отчество"
Binding="{Binding pat_patronymic}" Width="*/>
<DataGridTextColumn Header="Дата
рождения" Binding="{Binding birthdate,
StringFormat={ } {0:d MMMM yyyy 'г.' }}"
Width="*/>
<DataGridTextColumn Header="Номер
телефона" Binding="{Binding pat_phone_number}"
Width="*/>
</DataGrid.Columns>
</DataGrid>
<Button x:Name="RegisterPatientButton"
Content="Добавить пациента"
Click="RegisterPatientButton_Click"
HorizontalAlignment="Left" Margin="19,5,0,0"
Grid.Row="1" VerticalAlignment="Top" Height="29"
Width="169" Background="#FF629FE2"
BorderBrush="#FF629FE2"/>
<Button x:Name="EditPatientButton"
Content="Редактировать"
Click="EditPatientButton_Click"

```

```

HorizontalAlignment="Left" Margin="195,5,0,0"
Grid.Row="1" VerticalAlignment="Top" Height="29"
Width="169" Background="#FF629FE2"
BorderBrush="#FF629FE2"/>
    <Button x:Name="CreateOrderTB"
Content="Оформить заказ"
Click="CreateOrderTB_Click"
HorizontalAlignment="Left" Margin="371,5,0,0"
Grid.Row="1" VerticalAlignment="Top" Height="29"
Width="169" Background="#FF629FE2"
BorderBrush="#FF629FE2"/>
    <Button x:Name="SeeOrderTB"
Content="Посмотреть заказы"

```

```

Click="SeeOrderTB_Click"
HorizontalAlignment="Left" Margin="546,5,0,0"
Grid.Row="1" VerticalAlignment="Top" Height="29"
Width="169" Background="#FF629FE2"
BorderBrush="#FF629FE2" Visibility="Hidden"/>
</Grid>
</Window>

```

Весь код приложения находится на компакт-диске.

Компакт-диск с материалами проекта

На CD-диске располагается:

- Скрипт базы данных.
- Установщик программы.
- Проект программы.
- Файл курсового проекта в формате MS Word.
- Файл с кодом программы.