**Содержание**

Введение

1 Основание разработки

2 Назначение и область применения программы

3 Постановка задачи

3.1 Построение модели базы данных

3.2 Проектирование функциональных задач

3.3 Спецификация входных и выходных данных

4 Реализация программы

4.1 Анализ средств разработки

4.2 Обоснование выбора программного средства

4.3 Разработка интерфейса программного продукта

4.4 Проектирование основного и вспомогательного алгоритмов

4.5 Разработка программного модуля

Заключение

Библиография

Приложение

**Введение**

Электронная картотека — упорядоченное собрание данных, как правило на карточках или листах одинакового формата объединённых общим содержанием в электронном виде.

Актуальность применения электронных картотек заключается в том, что современные медицинские учреждения все чаще стремятся автоматизировать свои процессы, что позволяет значительно упростить и ускорить работу медицинского персонала, а также повысить качество и безопасность предоставляемых медицинских услуг.

Практическая значимость подобных приложений заключается в следующем:

1. Удобство хранения и доступа к медицинским данным пациентов. Электронная картотека позволяет хранить все данные пациента в одном месте, что облегчает их поиск и доступность для врачей и медицинского персонала.
2. Улучшение качества предоставляемых медицинских услуг. Благодаря электронной картотеке врачи имеют доступ к полной и актуальной информации о пациенте, что помогает им принимать обоснованные решения и предоставлять более качественное лечение.
3. Сокращение времени на обработку и анализ медицинских данных. Автоматизация процессов сбора, хранения и анализа данных позволяет сократить время, затрачиваемое на рутинные операции, и увеличить эффективность работы медицинского персонала.
4. Улучшение безопасности данных. Электронная картотека позволяет установить ограничения доступа к конфиденциальной информации и вести ее резервное копирование, что повышает безопасность медицинских данных пациентов.

Основой любой электронной системы является база данных.

База данных (БД) — это структурированная коллекция данных, которая используется для хранения информации о пациентах и их медицинской истории. Любая база данных создается при помощи систем управления базами данных.

Система управления базами данных (СУБД) — это специализированное программное обеспечение, предназначенное для управления и организации данных в базах данных.

Во время прохождения производственной практики в ГБУЗ «Районная больница г. Верхнеуральск», было дано задание, которое заключалось в разработке системы, которая позволит вести учет данных пациентов в электронном виде.

В соответствии с полученным заданием на производственной практике была определена цель курсового проекта: «Создание электронной картотеки пациента».

Исходя из цели работы был описан следующий круг задач курсового проекта:

1. Изучение требований заказчика и проведение анализа предметной области — данные пациента и их учет.
2. Проектирование модели базы данных.
3. Создание пользовательского интерфейса электронной картотеки.
4. Реализация функциональных возможностей электронной системы.
5. Тестирование и отладка приложения для обеспечения его корректной работы и соответствия требованиям.
6. Оформление технической документации в виде пояснительной записки на курсовой проект.

**1 Основание разработки**

Основанием для разработки электронной картотеки является задание, полученное на курсовой проект по профессиональному модулю ПМ.02 Осуществление интеграции программных модулей междисциплинарного курса МДК 02.01 Технология разработки программного обеспечения.

Руководителем курсового проекта является преподаватель специальных дисциплин, председатель цикловой методической комиссии специальности «Информационные системы и программирование» Брюханова Яна Олеговна.

Заказчиком на разработку электронной картотеки является ГБУЗ «Районная больница г. Верхнеуральск».

Требования заказчика заключалось в разработке программного приложения, предназначенного для ведения и учета данных о пациентах в электронном виде.

Преимуществами разработанного приложения являются:

* интуитивно-понятный пользовательский интерфейс;
* выполнение различных манипуляций с данными;
* электронное заполнение осмотра;
* осуществление входа с помощью одноразового кода;
* осуществления смены пароля с использованием подтверждения одноразового кода;
* формирование электронного осмотра в текстовом файле.

**2 Назначение и область применения программы**

Разработанное приложение предназначено для применения в сфере здравоохранения.

Система будет применяться для управления медицинскими данными пациентов, которая позволит улучшить качество и доступность медицинской помощи, оптимизировать процессы работы медицинских учреждений.

Разработанная программа является универсальной и может использоваться в работе любого медицинского учреждения.

Использование данного программного решения позволит выполнить следующую работу:

* осуществлять вход в систему под разными пользователями;
* осуществлять регистрацию пациента;
* осуществлять различные операции с данными (добавление, удаление, редактирование);
* создавать результаты осмотров в текстовом виде.

**3 Постановка задачи**

**3.1 Построение модели базы данных**

Модель базы данных разработана на основе анализа требований заказчика и изучения предметной области – данные пациента и их учет.

Модель базы данных – это абстрактная структура, которая определяет способ организации и хранения данных в базе данных. Она определяет структуру данных, схему таблиц, связей между таблицами и правила целостности данных. Модель базы данных помогает описать и организовать данные для их эффективного использования и обработки. В зависимости от потребностей и особенностей проекта, могут быть использованы различные модели баз данных, такие как реляционная, иерархическая, сетевая, объектно-ориентированная, графовая и др.

Предметная область – область знаний или деятельности, которая имеет определенные характеристики, термины, понятия, методы и принципы, связанные с определенной областью знания или практики. Предметная область может быть узкой (например, физика элементарных частиц) или широкой (например, медицина) и обычно определяет предмет исследования или работы в определенной области. Например, в области информационных технологий предметная область может включать такие области, как программирование, базы данных, сети, компьютерная безопасность и т. д.

В разработанной модели базы данных объектами являются созданные таблицы:

Следует отметить то, что каждый из вышеперечисленных объектов (таблиц) имеет свой атрибут (значение):

1. Препараты (Medication) – обеспечивает хранение информации о препарате с коротким описанием.
2. Жалобы (Complaints) – обеспечивает хранение информации о жалобах пациентов с препаратами для лечения.
3. Осмотр (Exam) – обеспечивает хранение информации об проведенном осмотре пациента.
4. Диагноз (Diagnosis) – обеспечивает хранение информации о диагнозе.
5. Пол (Gender) – обеспечивает хранение информации о поле человека.
6. Посещения (Visiting) – обеспечивает хранение информации о посещениях пациента, его жалобах и проведенном осмотре.
7. Пациент (Patient) – обеспечивает хранение информации данных пациента, которые используются для осуществления регистрации на осмотр.
8. Прием (Reception) – обеспечивает хранение информации о дате приема и о враче, который обеспечивал осмотр.
9. Роль (Roles) – обеспечивает хранение информации о ролях пользователей.
10. Специальность (Specialization) – обеспечивает хранение информации о разных специальностях, которыми наделяются пользователи.
11. Расписание (Schedule) – обеспечивает хранение информации о расписание работы врачей.
12. Доктор (Doctor) – обеспечивает хранение информации о враче, его отделение, специальности, расписание и многое другое.
13. Пользователи (Users) – обеспечивает хранение информации для осуществления входа (логин и пароль), а также ФИО входящего пользователя.
14. Отделение (Department) – обеспечивает хранение информации об отделе, а именно его название и адрес, где он находится.

Модель базы данных представлена в виде ER-модели (рисунок 1), которая была создана в Microsoft Visio

ER-модель, представленная на рисунке 1, позволяет описать предметную область приложения; сущности базы данных и связи между ними.

ER-диаграмма – это визуальная модель, используемая для описания структуры и взаимодействия между сущностями (сущности-связи) в системе. ER-диаграмма широко используется в области проектирования баз данных для определения сущностей, их атрибутов и связей между ними. Она позволяет разработчикам легко понять структуру данных и связи между ними, что помогает в создании эффективной базы данных. ER-диаграмма состоит из сущностей (объектов), атрибутов (свойств объектов) и связей (отношений между объектами). Визуальные символы, такие как овалы, прямоугольники и линии, используются для представления этих элементов.

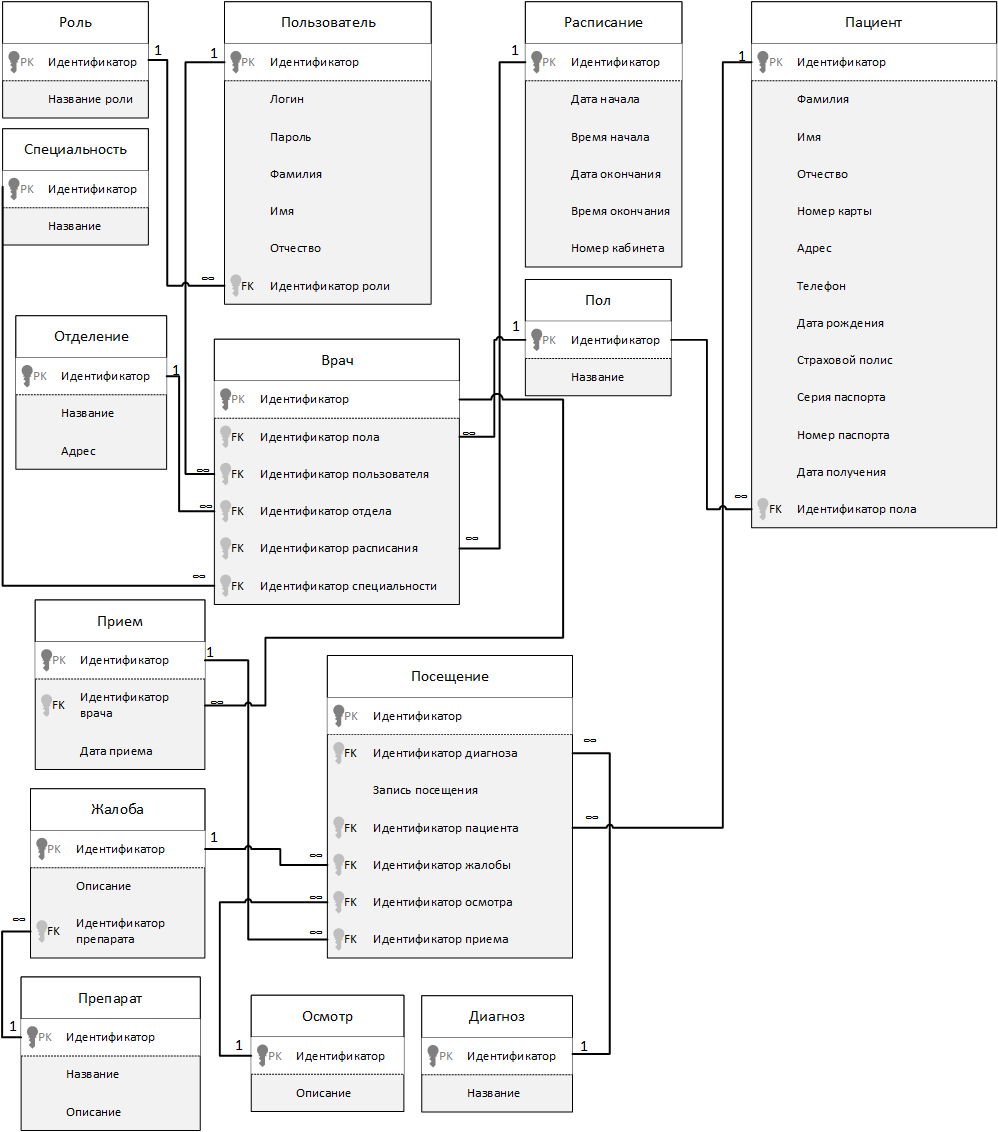


Рисунок 1 – Модель базы данных

Вышеперечисленные объекты (таблицы) были созданы в системе управления базами данных Microsoft SQL Server и являются исходными таблицами.

На рисунке 2 представлена физическая модель базы данных:

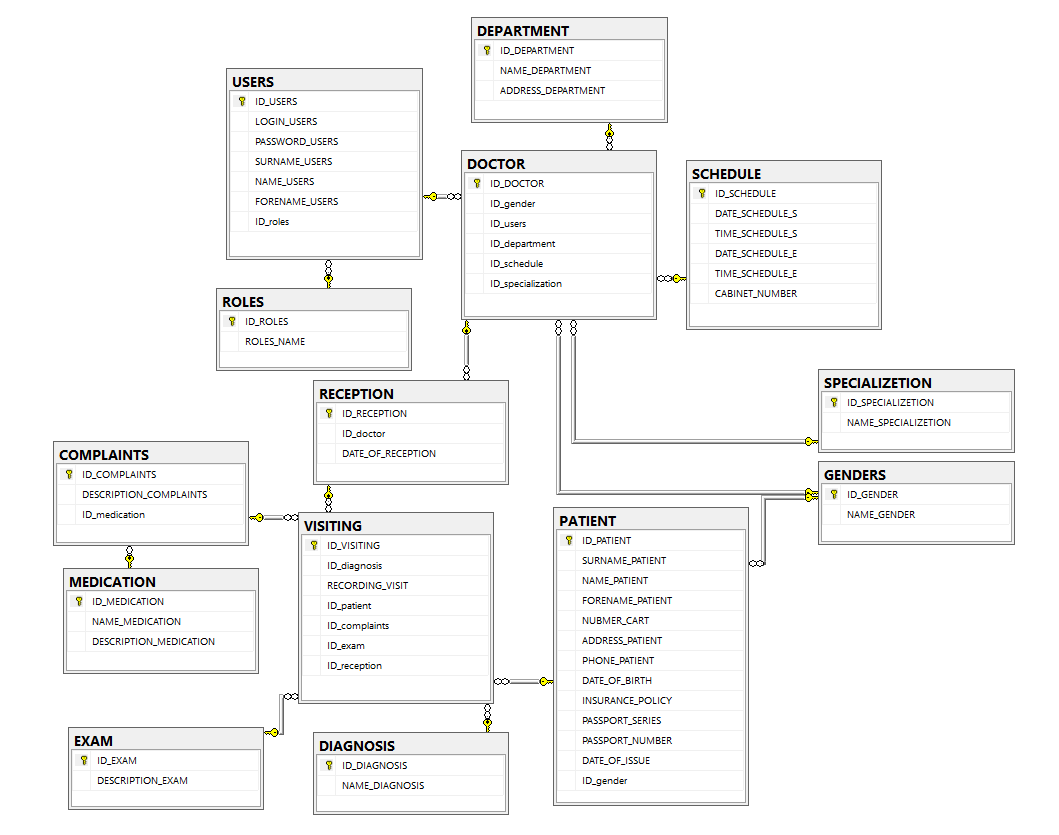


Рисунок 2 – Физическая модель базы данных

Далее приводится описание всех зависимостей между основными сущностями с подробным их описанием.

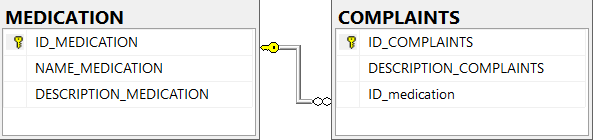


Рисунок 3 – Связь «Medication (препарат)» с «Complaints (жалобы)»

Таблицы «Препарат» и «Жалобы» находятся в отношение «один-ко- многим», так как один препарат может применяться в нескольких случаях.



Рисунок 4 – Связь «Roles (роли)» с «Users (пользователи)»

Таблицы «Роли» и «Пользователи» находятся в отношение «один-ко- многим», так как множество пользователей могут иметь одинаковую роль, для работы в системе.

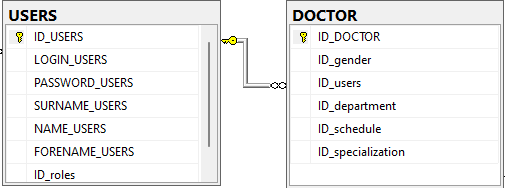


Рисунок 5 – Связь «Users (пользователи)» с «Doctor (врач)»

Таблицы «Пользователи» и «Врач» находятся в отношение «один-ко- многим», так как осуществление входа врача в систему используются заранее выданные уникальные данные.



Рисунок 6 – Связь «Doctor (врач)» с «Schedule (расписание)»

Таблицы «Врач» и «Расписание» находятся в отношение «один-ко- многим», так как у многих врачей может быть одинаковое расписание.

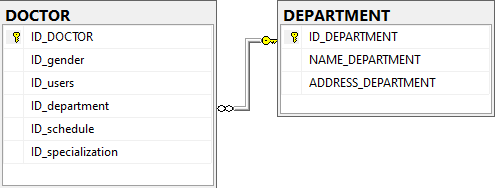


Рисунок 7 – Связь «Doctor (врач)» с «Department (отделение)»

Таблицы «Врач» и «Отделение» находятся в отношение «один-ко- многим», так как в одном отделение может работать несколько врачей.

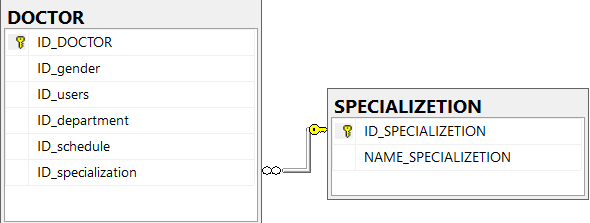


Рисунок 8 – Связь «Doctor (врач)» с «Specialization (специализация)»

Таблицы «Врач» и «Специализация» находятся в отношение «один-ко- многим», так как одна специализация может быть у нескольких врачей.

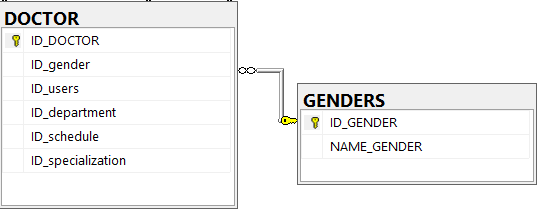


Рисунок 9 – Связь «Doctor (врач)» с «Gender (пол)»

Таблицы «Врач» и «Пол» находятся в отношение «один-ко-многим», так как у множества врачей может быть одинаковый пол.

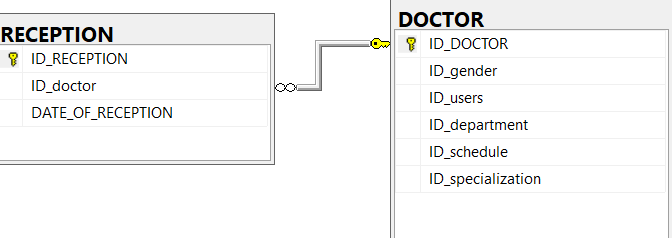


Рисунок 10 – Связь «Reception (прием)» с «Doctor (врач)»

Таблицы «Прием» и «Врач» находятся в отношение «один-ко-многим», так как один врач может проводить несколько приемов.

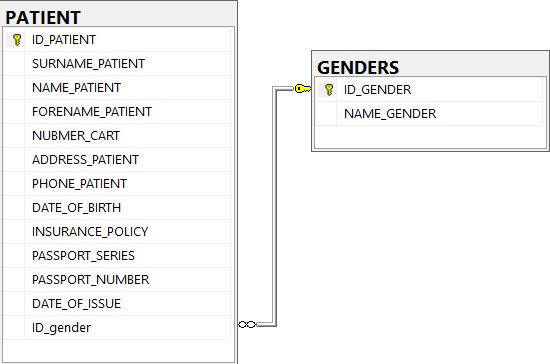


Рисунок 11 – Связь «Patient (пациент)» с «Gender (пол)»

Таблицы «Пациент» и «Пол» находятся в отношение «один-ко- многим», так как один и тот же пол может быть у нескольких пациентов.

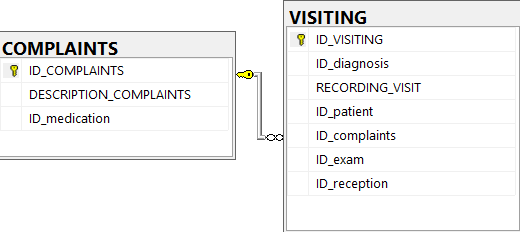


Рисунок 12 – Связь «Complaints (жалобы)» с «Visiting (посещение)»

Таблицы «Жалобы» и «Посещение» находятся в отношение «один-ко- многим», так как множество жалоб фиксируется при одном посещении.

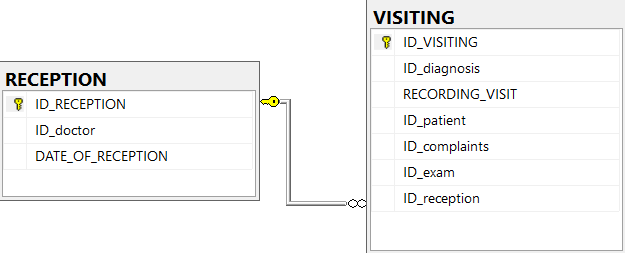


Рисунок 13 – Связь «Reception (прием)» с «Visiting (посещение)»

Таблицы «Прием» и «Посещение» находятся в отношение «один-ко- многим», так как один прием может проводиться при неоднократном посещение врача.

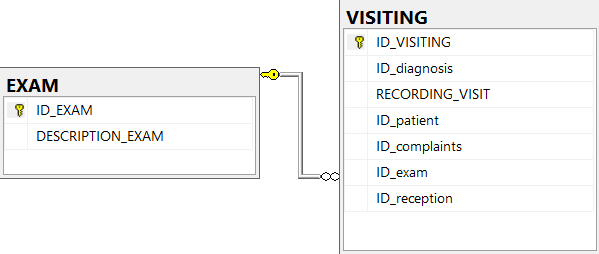


Рисунок 14 – Связь «Exam (осмотр)» с «Visiting (посещение)

Таблицы «Осмотр» и «Посещение» находятся в отношение «один-ко- многим», так как один осмотр может формироваться при неоднократном посещение врача пациентом.

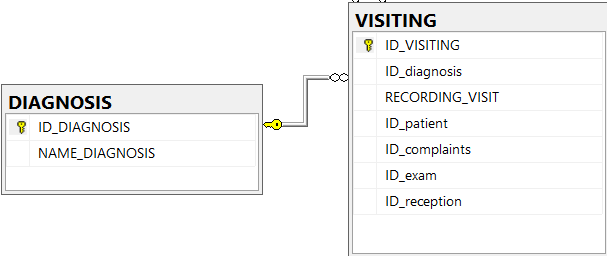


Рисунок 15 – Связь «Diagnosis (диагноз)» с «Visiting (посещение)»

Таблицы «Диагноз» и «Посещение» находится в отношение «один-ко- многим», так как один и тот-же диагноз может быть поставлен при одном и том же посещение врача.

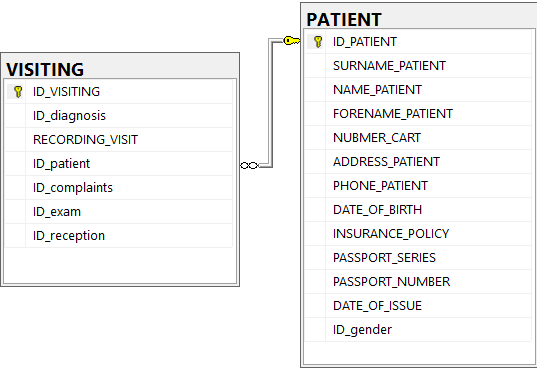


Рисунок 16 – Связь «Visiting (посещение)» с «Patient (пациент)»

Таблицы «Посещение» с «Пациент» находятся в отношение «один-ко- многим», так как один пациент может посещать одного и того же врача.

**3.2 Проектирование функциональных задач**

Разработанное электронная картотека пациента обладает следующими функциональными возможностями:

* авторизация в систему под своей учетной записью — нужно для предотвращение постороннего входа в систему;
* осуществления входа по одноразовому коду, отправленному на почту;
* осуществления смены пароля путем подтверждения одноразового кода;
* осуществление операций по добавлению, изменению и удалению пациентов в системе — позволяет учитывать и обновлять личные данные пациентов;
* возможность блокировки приложения с сохранением данных на окнах — осуществляет скрытие окон, которые имеют возможность для работы с данными пациентов.
* осуществление операций по добавлению, изменению и удалению осмотра пациента - позволяет учитывать и обновлять данные осмотров пациентов;
* осуществлять поиск пациентов, прошедших осмотр у врача;
* создание электронного осмотра пациента — нужно для предоставления осмотра в удобном формате для пациента;
* осуществлять поиск зарегистрированных пациентов в системе по отображаемым данным.

**3.3 Спецификация входных и выходных данных**

Входная информация – это информация, которая подается на вход в систему, процесс или программу. Она может быть представлена в виде текста, чисел, изображений, звуковых файлов и так далее. Входная информация служит для выполнения определенных операций, вычислений или обработки данных.

Входные данные разработанного приложения: логин и пароль, которые осуществляют вход в саму систему, данные о пациенте (ФИО, паспортные данные, полис и пол), которые необходимы для регистрации пациента в системе, а также данные вводимые пользователем при осмотре определенного пациента.

Таблица 1 – «Medication» (препарат)

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Комментарий |
| NAME\_MEDICATION | Nvarchar(60) | Название препарата |
| DESCRIPTION\_MEDICATION | Nvarchar(300) | Описание препарата |

Таблица 2 – «Roles» (роли)

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Комментарий |
| ROLES\_NAME | Nvarchar(50) | Название роли |

Таблица 3 – «Users» (пользователи)

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Комментарий |
| LOGIN\_USERS | Nvarchar(30) | Логин |
| PASSWORD\_USERS | Nvarchar(30) | Пароль |
| SURNAME\_USERS | Nvarchar(30) | Фамилия |
| NAME\_USERS | Nvarchar(30) | Имя |
| FORENAME\_USERS | Nvarchar(30) | Отчество |

Таблица 4– «SCHEDULE» (расписание)

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Комментарий |
| DATE\_SCHEDULE\_S | Date | Дата начала |
| TIME\_SCHEDULE\_S | Time(7) | Время начала |
| DATE\_SCHEDULE\_E | Date | Дата окончания |
| TIME\_SCHEDULE\_E | Time(7) | Время окончания |
| CABINET\_NUMBER | Integer | Номер кабинета |

Таблица 5 – «Department» (отделение)

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Комментарий |
| NAME\_ DEPARTMENT | Nvarchar(50) | Название отдела |
| ADDRESS\_ DEPARTMENT | Nvarchar(100) | Адрес отдела |

Таблица 6 – «Specialization» (специальность)

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Комментарий |
| NAME\_SPECIALIZETION | Nvarchar(50) | Название специальности |

Таблица 7 – «Genders» (пол)

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Комментарий |
| NAME\_GENDER | Nvarchar(7) | Название пола |

Таблица 8 – «Complaints» (жалобы)

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Комментарий |
| DESCRIPTION\_COMPLAINTS | Nvarchar(200) | Описание жалобы |

Таблица 9– «Reception» (прием)

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Комментарий |
| DATE\_OF\_RECEPTION | Date | Дата приема |

Таблица 10 – «Exam» (осмотр)

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Комментарий |
| DESCRIPTION\_EXAM | Nvarchar(MAX) | Описание осмотра |

Таблица 11 – «Diagnosis» (диагноз)

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Комментарий |
| NAME\_DIAGNOSIS | Nvarchar(60) | Название диагноза |

Таблица 12– «Patient» (пациент)

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Комментарий |
| SURNAME\_PATIENT | Nvarchar(50) | Фамилия |
| NAME PATIENT | Nvarchar(50) | Имя |
| FORENAME PATIENT | Nvarchar(50) | Отчество |
| NUBMER CART | Integer | Номер карты |
| ADDRESS PATIENT | Nvarchar(100) | Адрес |
| PHONE PATIENT | Nvarchar(20) | Телефон |
| DATE OF BIRTH | Date | Дата рождения |
| INSURANCE POLICY | Nvarchar(19) | Страховой полис |
| PASSPORT SERIES | Nvarchar(5) | Серия паспорта |
| PASSPORT NUMBER | Nvarchar(6) | Номер паспорта |
| DATE OF ISSUE | Date | Дата выдачи |

Таблица 13– «Visiting» (посещение)

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Комментарий |
| RECORDING VISIT | Nvarchar(100) | Запись посещения |

Выходная информация – это информация, которая предоставляется или передается системой или устройством наружу или другой системе или устройству. Она может быть представлена в различных формах, таких как текст, изображения, звук или видео. Выходная информация может быть результатом обработки или анализа входной информации или действий, выполненных системой или устройством. Она может использоваться для передачи данных, отображения информации пользователю или выполнения других задач и функций. Например, вывод данных о пациенте (рисунок 17) и (рисунок 18), также формирование электронного осмотра (рисунок 19).

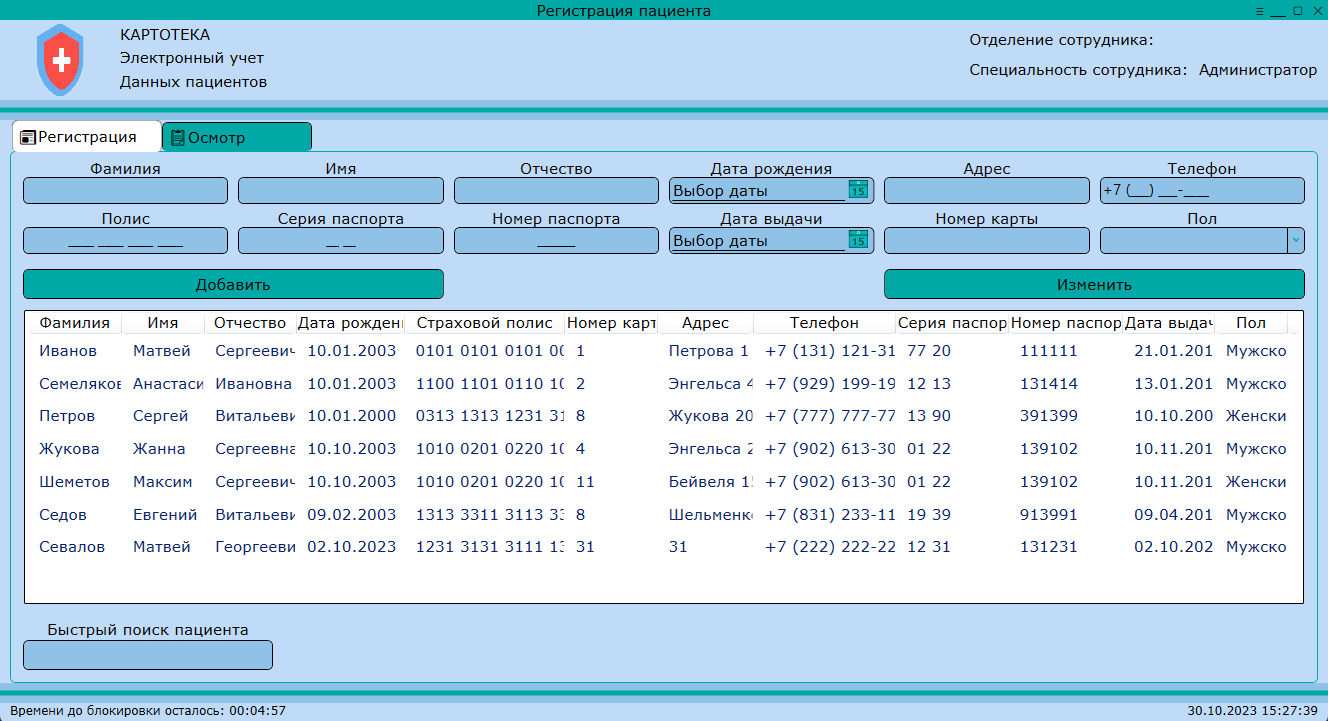


Рисунок 17 – вывод данных пациентов

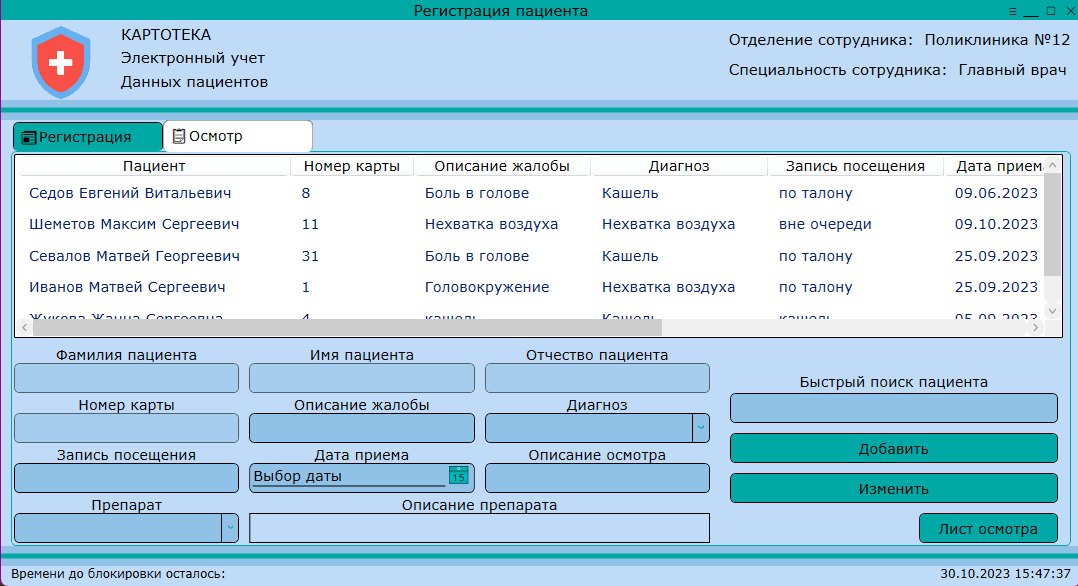


Рисунок 18 – вывод данных осмотра пациентов

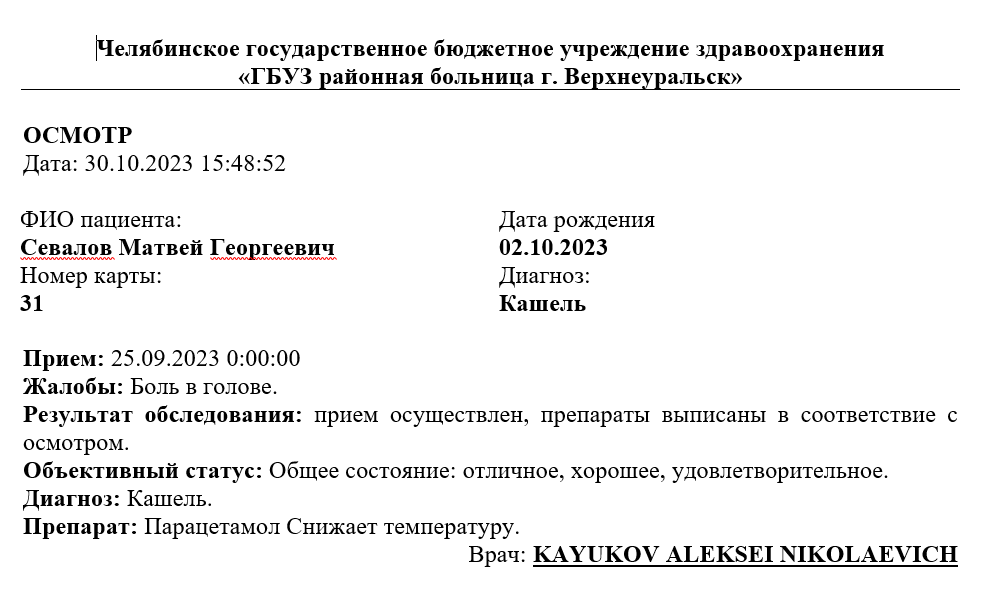


Рисунок 19 – электронный осмотр пациента

**4 Реализация программы**

**4.1 Анализ средств разработки**

1. Аналогом сред для разработки ПО можно выделить следующие среды разработки:

1. Eclipse - это среда разработки программного обеспечения, разработанная на Java. Она позволяет разрабатывать приложения на разных языках программирования, таких как Java, C++, Python и многих других. Eclipse имеет широкий набор инструментов и плагинов, которые облегчают процесс разработки и улучшают продуктивность разработчика.

2. IntelliJ IDEA - это интегрированная среда разработки программного обеспечения, созданная для разработчиков на языке Java. Она предоставляет широкий набор инструментов, таких как автодополнение кода, рефакторинг, отладка, а также поддержку для других языков программирования, таких как Kotlin, Groovy и Scala. IntelliJ IDEA также предлагает мощные инструменты для разработки мобильных приложений.

3. NetBeans - это свободная интегрированная среда разработки программного обеспечения, поддерживающая несколько языков программирования, включая Java, C++, PHP и другие. Она обладает удобным пользовательским интерфейсом и предлагает инструменты для разработки веб-приложений, мобильных приложений и десктопных приложений. NetBeans также имеет широкий набор плагинов, которые позволяют расширять функциональность среды разработки.

2) Языки программирования, на котором можно было разработать ПО следующие:

1. Java - это объектно-ориентированный язык программирования, разработанный компанией Sun Microsystems. Он известен своей платформой Java, которая обеспечивает кросс-платформенность и портативность кода.

Характеристики:

- компилируется в байт-код, который выполняется на виртуальной машине Java (JVM);

- обладает сильной типизацией и автоматическим управлением памятью;

- поддерживает множество библиотек и фреймворков для разработки различных типов приложений;

- имеет широкое применение в веб-разработке, мобильной разработке, разработке корпоративных решений и других областях.

2. C++ - это язык программирования, разработанный на основе языка C. Он является мощным и гибким языком, способным выполнять широкий спектр задач, от системного программирования до разработки игр.

Характеристики:

- компилируемый язык программирования, который генерирует машинный код;

- обладает низкоуровневыми возможностями, позволяющими получать прямой доступ к памяти и манипулировать битами;

- поддерживает классы и объектно-ориентированное программирование, а также приводит к яве компиляции (templates);

- широко используется в разработке игр, системного программирования, высокоэффективных приложений и других высоконагруженных систем.

3) СУБД, на котором можно было бы разработать ПО следующие:

1. Oracle Database - это реляционная система управления базами данных (СУБД), разработанная корпорацией Oracle. Она широко используется в крупных корпоративных системах и предлагает высокую производительность и надежность. Характеристики: масштабируемость, высокая производительность, поддержка сложных структур данных, автоматическая репликация, безопасность и аудит данных.

2. MySQL - это открытая реляционная СУБД, разработанная и поддерживаемая компанией Oracle. Она популярна среди разработчиков веб-приложений и отличается простотой использования и гибкостью. Характеристики: открытый исходный код, масштабируемость, высокая производительность, простота использования, поддержка транзакций и отказоустойчивость.

3. SQLite - это встроенная СУБД, которая не требует отдельного сервера и устанавливается как библиотека внутри приложений. Она широко используется в мобильных приложениях и небольших проектах. Характеристики: небольшой размер, простота использования, не требует отдельного сервера, хранение базы данных в одном файле, свободная лицензия.

4. PostgreSQL - это объектно-реляционная СУБД с открытым исходным кодом. Она предлагает расширенные возможности и большую гибкость в сравнении с некоторыми другими СУБД. Характеристики: открытый исходный код, поддержка сложных структур данных, обработка графических и географических данных, поддержка транзакций, масштабируемость.

4) Технологии доступа к данным, которые можно было использовать следующие:

1. MySQL Workbench: Это инструмент управления базами данных MySQL, который предоставляет возможность соединиться с базой данных, создавать и изменять структуру таблиц, выполнять запросы и многое другое. Он имеет графический интерфейс пользователя, который делает его более простым в использовании для новичков. Однако MySQL Workbench не поддерживает такие функции, как схемы, вложенные транзакции и экспорт в формат Microsoft SQL Server.

2. Oracle SQL Developer: Это бесплатное инструментарий для разработки и управления базами данных Oracle. Он позволяет создавать и редактировать объекты базы данных, выполнять SQL-запросы и отчеты, а также отслеживать производительность базы данных. Oracle SQL Developer имеет обширные функции, такие как поддержка экспорта и импорта данных, отладчик SQL, моделирование данных и инструменты для рабочей среды разработчика.

3. DBeaver - это мощный инструмент для управления базами данных, разработанный для различных ОС, таких как Windows, macOS и Linux. Он поддерживает большое количество СУБД, включая MySQL, PostgreSQL, Oracle, SQL Server, SQLite и многие другие. DBEAVER предлагает пользователю интуитивно понятный интерфейс с широким спектром функций, которые позволяют управлять и администрировать базы данных.

**4.2 Обоснование выбора программного средства**

Для разработки электронной картотеки была выбрана среда разработки Visual Studio. Выбор пал именно на эту среду, потому что у меня имеется опыт в работе с ней, а также большое количество информации в интернете, которая позволяет решить какие-либо вопросы в реализации различных операций. Данная среда поддерживает нужный мне язык программирования и файлы совместимые с системными. Удобный интерфейс, а также проведение тестирования с использованием точек остановок.

СУБД была выбрана MsSQL так как она имеют высокую скорость обработки запросов и выполнения операций. Поддерживает различные языки программирования, а также имеет мощный набор инструментов разработки и управления базами данных. Также выбрал ее, потому что она хорошо интегрируется с проектами от Microsoft, к примеру .Net Framework.

Технология доступа к данным пала на MSSMS, потому что я имею опыт в работе с ней. Также она обладает высокой надежностью, удобным интерфейсом, большим набором инструментов.

Язык программирования использовал C#, потому что имеется опыт в работе с ним, а также он легок в изучение. Наличие обширной документации на сайтах, которые помогают при возникновении вопросов. Он также имеет интеграцию с платформой .NET и обладает высокой надежностью и много поточностью.

**4.3 Разработка интерфейса программного продукта**

Работа с электронной картотекой начинается с запуска загрузочной формы (рисунок 20), после которого осуществляется переход на окно авторизации (рисунок 21).



Рисунок 20 – Окно запуска системы

В окне авторизации расположены следующие компоненты:

* кнопки свернуть и закрыть окно;
* текстовые поля для ввода логина и пароля пользователя;
* текстовая ссылка для входа с помощью одноразового кода;
* кнопка «Авторизоваться»;
* текстовая ссылка для осуществления изменения пароля.

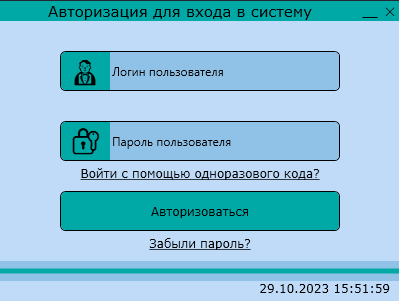


Рисунок 21 – Окно авторизации в систему

При нажатии на кнопку «Авторизоваться» происходит проверка корректности ввода логина и пароля пользователя, осуществляется переход на главное окно пользователя.

При неудачном вводе данных происходит уведомление пользователя о том, что он совершил ошибку (рисунок 22).

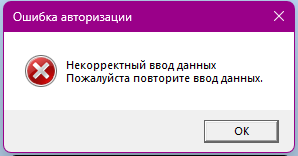


Рисунок 22 – ошибка при авторизации пользователя

Для осуществления входа с помощью одноразового кода необходимо ввести ваш логин в соответствующее поле, а также осуществить нажатие на текстовую ссылку «Войти с помощью одноразового кода?», после чего появиться уведомление о том, что код был выслан на указанную почту (рисунок 23), затем вводим код в поле для пароля и нажимаем на кнопку «Авторизоваться» и попадаем на главное окно системы.

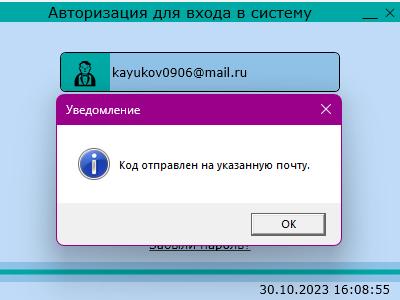


Рисунок 23 – уведомление об отправке одноразового кода

Если пользователь по каким-либо причинам забыл пароль, то он может изменить его. Для этого необходимо ввести логин пользователя, а также осуществить нажатие по текстовой ссылке «Забыли пароль?», после чего произойдет уведомление пользователя, о том, что на его почту был выслан одноразовый код (рисунок 23), а после происходит переход на окно «Восстановления пароля» (рисунок 24).

Окно восстановления пароля содержит текстовые поля для ввода кода подтверждения, нового пароля и повтора пароля, а также кнопки для подтверждения изменения пароля, сворачивания и закрытия окна.

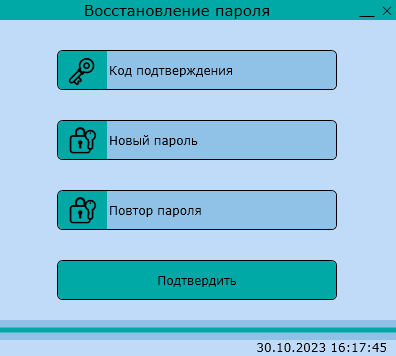


Рисунок 24 – окно восстановления пароля.

После нажатия на кнопку «Подтвердить» происходит проверка, что код подтверждения соответствует верному (рисунок 25), а также поля ввода пароля соответствуют друг другу (рисунок 26).

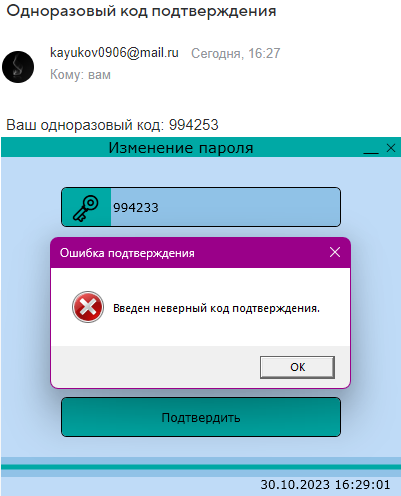


Рисунок 25 – ошибка соответствия кода подтверждения

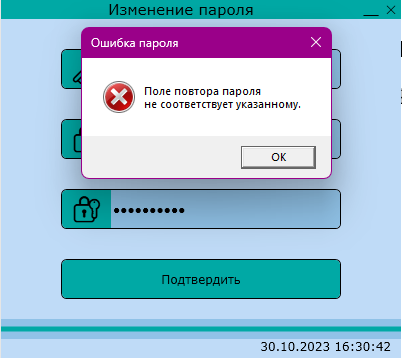


Рисунок 26 – ошибка соответствия паролей

Как только пользователь корректно ввел все данные и осуществил нажатие на кнопку «Подтвердить» - произойдет изменение пароля.

Осуществление входа на главное окно происходит после верного указывания одной из комбинаций: логина и пароля, либо логина и одноразового кода (рисунок 27).

Главное окно системы содержит следующие компоненты:

* меню с дополнительными функциями (рисунок 28);
* кнопки, осуществляющие сворачивание, разворачивание и закрытие окна;
* кнопки для перехода в раздел осмотра пациента (рисунок 29);
* поля ввода данных, выбора даты, а также пола пациента для ввода нужных данных для осуществления регистрации;
* кнопки, осуществляющие добавление и изменение данных;
* контекстное меню для управления данными в списке вывода пациентов;
* текстовое поля для осуществления поиска по всем данным.

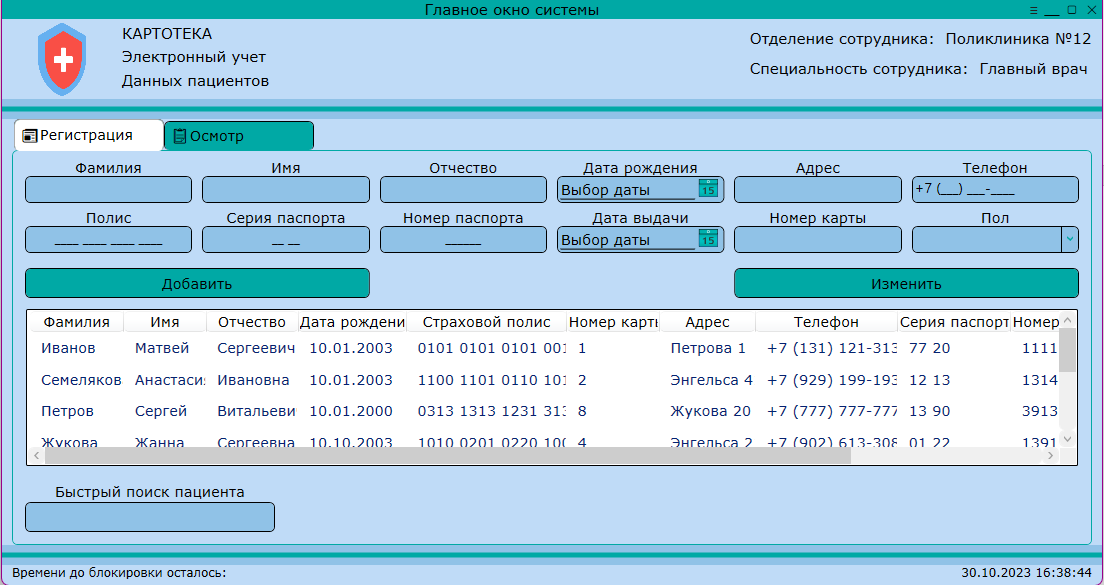


Рисунок 27 – главное окно системы

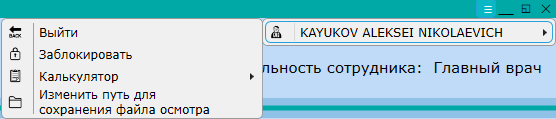


Рисунок 28 – меню с дополнительным функционалом

Меню содержит следующие возможности:

* выход на окно авторизации по нажатию: «Выйти» (рисунок 21);
* блокировки главного меню по нажатию: «Заблокировать» (рисунок 30, 31);
* открытие калькулятора при наведении на «Калькулятор» (рисунок 33);
* изменение пути для сохранения файла осмотра пациента по нажатию: изменить путь для сохранения файла осмотра».

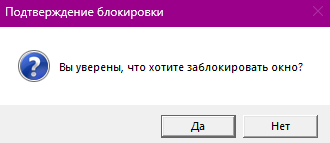


Рисунок 30 – подтверждение блокировки



Рисунок 31 – окно блокировки главного окна

Окно блокировки содержит поле для ввода пароля пользователя и кнопку «Войти». По нажатию кнопки происходит проверка того, что пароль соответствует текущему пользователю, если нет появиться сообщение об ошибке (рисунок 32), иначе произойдет переход на главное окно системы (рисунок 27).

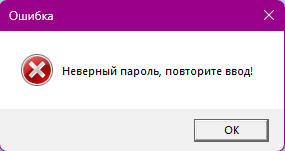


Рисунок 32– ошибка неверного ввода пароля

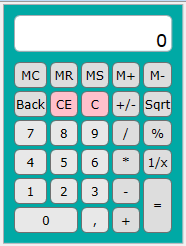


Рисунок 33 – открытие калькулятора

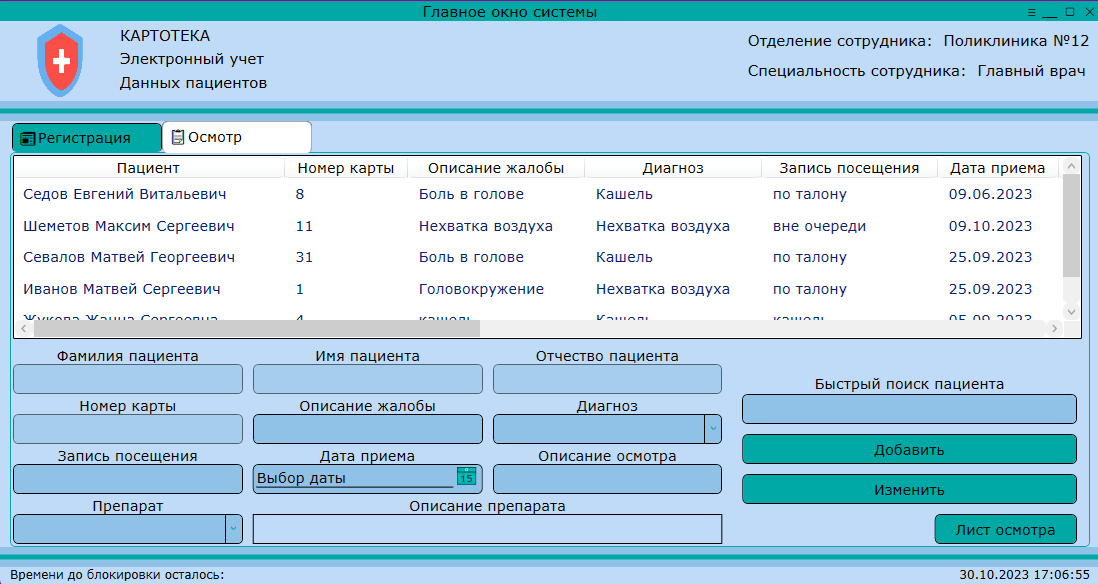


Рисунок 29 – переход на раздел осмотра пациентов

Для того чтобы осуществить добавление пациента необходимо верно указать данные и осуществить нажатие на кнопку «Добавить», после чего произойдет проверка введённых данных пользователем на корректность и само добавление пациента (рисунок 34), иначе появиться сообщение об ошибки ввода данных (рисунок 35)

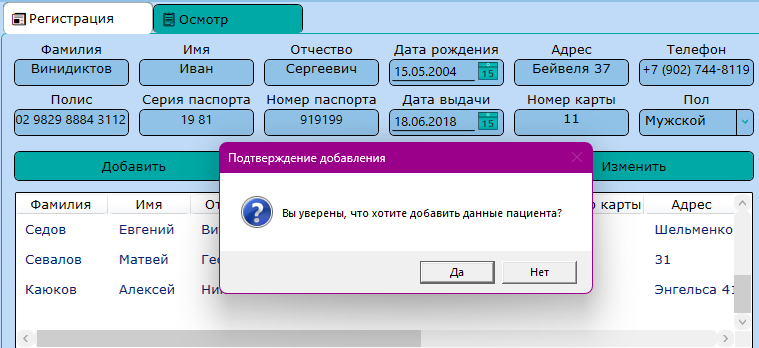


Рисунок 34 – подтверждение добавления данных пациента

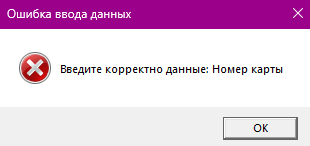


Рисунок 35 – сообщение об неверном указание номера карты пациента

Для осуществления изменения данных пациента необходимо выбрать нужного пациента, а после чего вызвать контекстное меню нажатием правой кнопкой мыши и выбрать пункт «Изменить», после чего изменяем нужные нам данные и осуществляем нажатие на кнопку «Изменить» (рисунок 36).

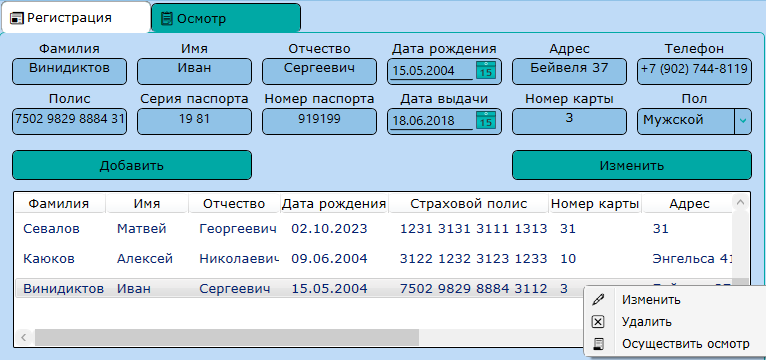


Рисунок 36 – изменение данных пациента

Для осуществления удаления данных пациента необходимо выбрать нужного пациента, а после чего вызвать контекстное меню нажатием правой кнопкой мыши и выбрать пункт «Удалить» (рисунок 36).

Осуществление поиска нужного пациента происходит с помощью ввода данных об пациенте в текстовое поле «Быстрый поиск пациента» (рисунок 37).

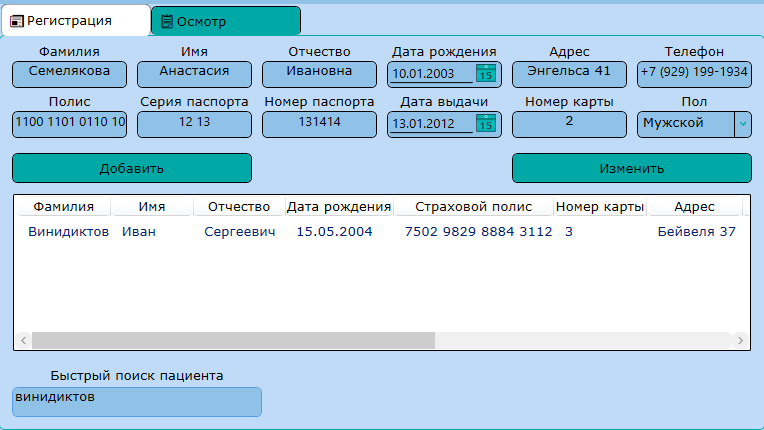


Рисунок 37 – поиск пациента

Проведение осмотра происходит путем выбора нужно пациента и осуществление вызова контекстного меню по нажатию правой кнопки мыши и выбора пункта «Осуществить осмотр» (рисунок 36). После чего необходимо перейти в раздел «Осмотр» (рисунок 29), в котором появиться выбранный пациент. Вносим данные об осмотре пациента, после чего нажимаем «Добавить» (рисунок 38).

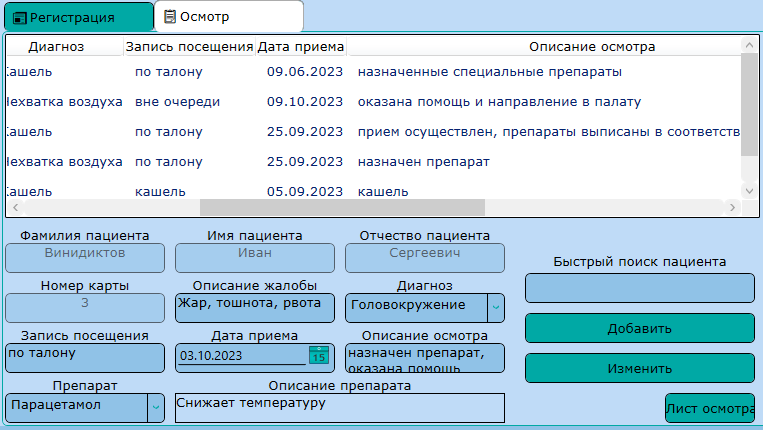


Рисунок 38 – добавление осмотра пациента

Изменение осмотра, удаления и поиска пациента происходит точно также, как и в разделе «Регистрация».

Добавление данных, выбираемых из раскрывающегося списка необходимо выбрать нужный список и в самом низу указать параметры после чего нажать на кнопку «+» (рисунок 39).

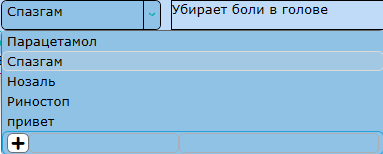


Рисунок 39 – добавление данных в раскрывающийся список

Осуществление создания электронного осмотра происходит путем нажатия по нужно пациенту в разделе «Осмотр» и вызова контекстного меню нажатием правой кнопки мыши с выбор параметра «Сформировать осмотр», после чего вызовется диалоговое окно сохранения (рисунок 39).

После чего выбранный нами путь для сохранения осмотра запомниться и позволит не вызывать каждый раз диалоговое окно. Для изменения пути сохранения файла в меню дополнительных функций (рисунок 28) необходимо выбрать параметр «Изменить путь для сохранения файла осмотра», которое позволит при дальнейшем формирование осмотра вызвать диалоговое окно для смены пути сохранения осмотра.

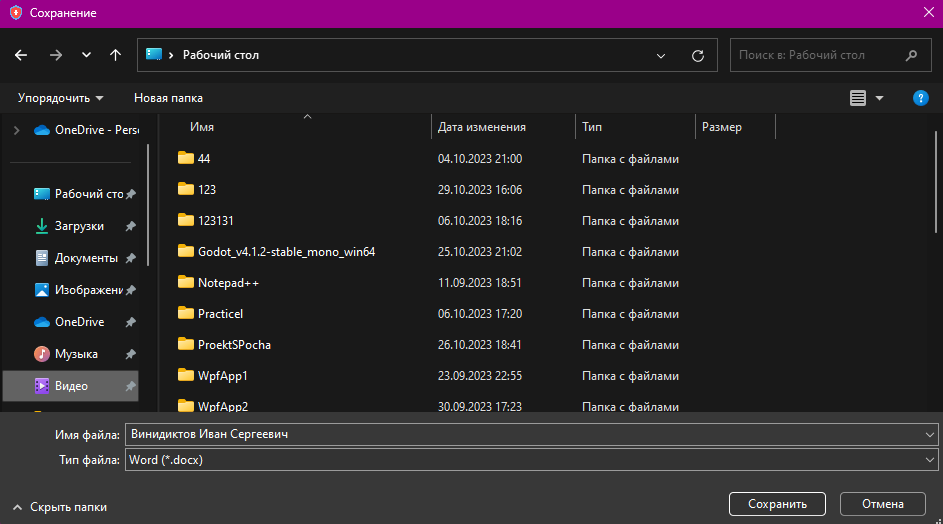


Рисунок 40 – диалоговое окно сохранения осмотра

**4.4 Проектирование основного и вспомогательного алгоритмов**

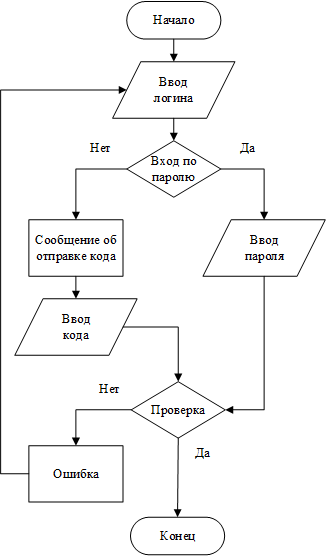
Алгоритм - это логические инструкции или шаги, которые определяют порядок выполнения определенной задачи или решения определенной проблемы в программном обеспечении. Он описывает, каким образом должны быть выполнены определенные действия или операции, чтобы достичь желаемого результата.

Алгоритмы ПО являются основой для разработки программ и являются важным компонентом в различных областях, таких как компьютерная наука, информационные технологии, искусственный интеллект, анализ данных и других.

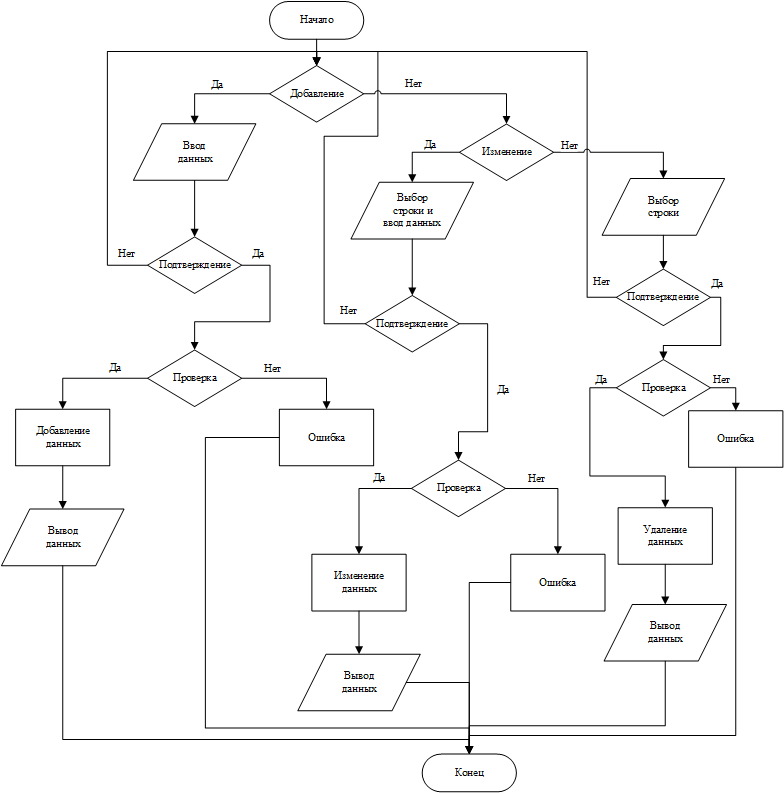
Основные функции программы:

* авторизация в систему (блок-схема 1);
* осуществление операций по добавлению, изменению, удалению данных регистрации пациента (блок-схема 2);
* осуществление электронного осмотра (блок-схема 3).

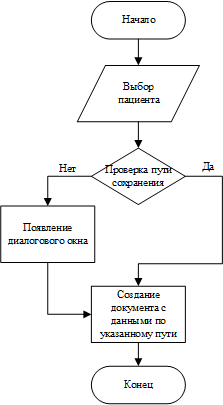
Блок-схема 1 — Авторизация в систему



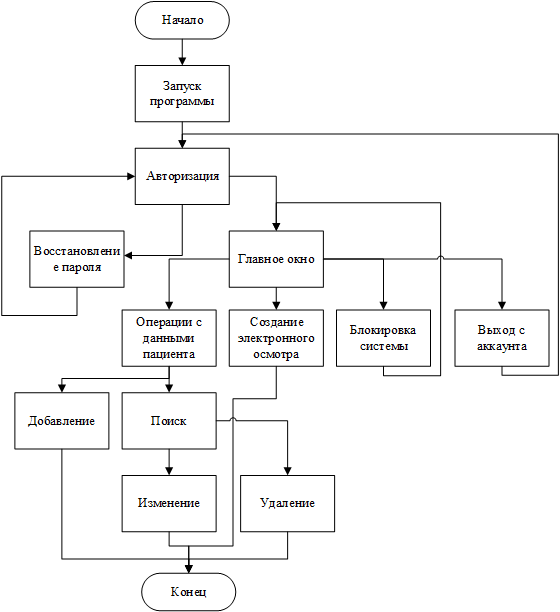
Блок-схема 2 – Операции для работы с данными пациента



Блок-схема 3 – Осуществление электронного осмотра



Блок-схема 4 – Логика работы программы



**4.5 Разработка программного модуля**

Программный модуль – это отдельная часть программного кода, фрагмент программы, который полностью самостоятельно выполняет свою задачу, оформленный в виде отдельного файла с исходным кодом или поименованной непрерывной её части.

В приложение по учету электронных данных пациента были реализованы следующие функции:

* загрузочное окно программы (рисунок 40);
* авторизация в систему (рисунок 41);
* восстановления пароля (рисунок 42);
* блокировка системы (рисунок 43);
* операции по добавлению, изменению и удалению данных пациента (рисунок 44);
* поиск пациента (рисунок 45);
* изменение пути сохранения файла осмотра (рисунок 46);
* формирование электронного осмотра (рисунок 47).

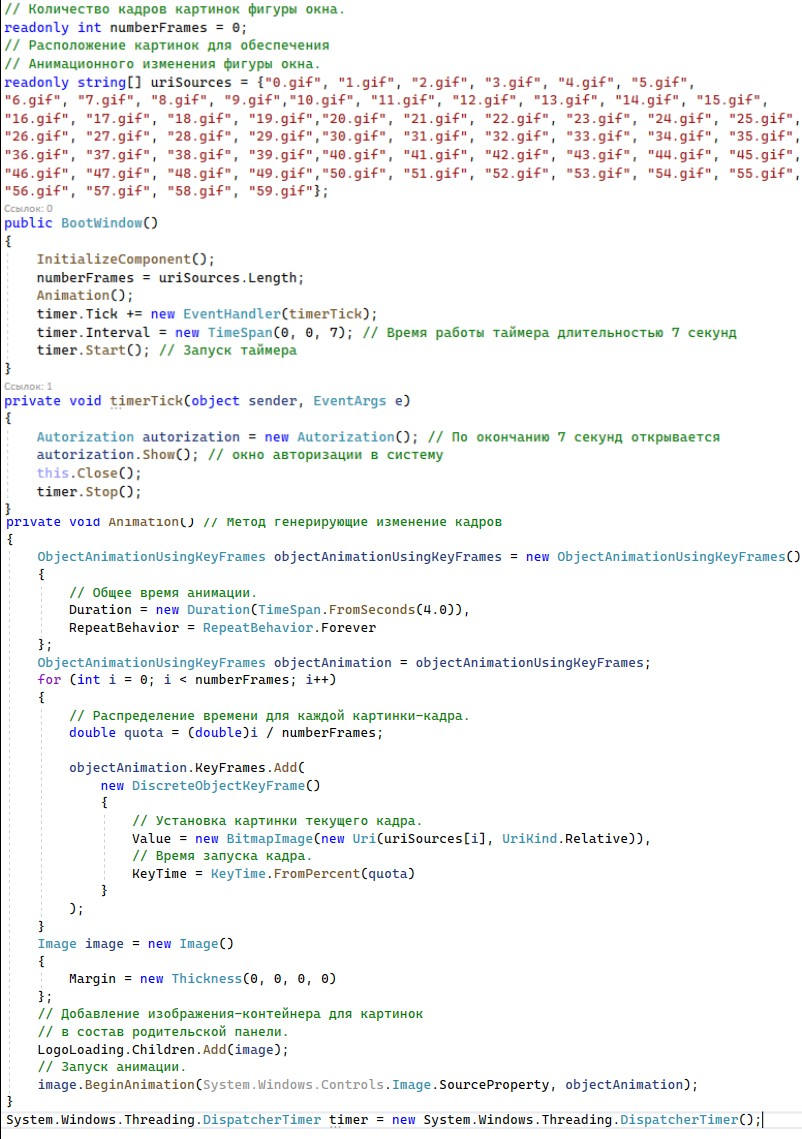


Рисунок 40 – фрагмент кода загрузочного окна



Рисунок 41 – фрагмент кода окна авторизации

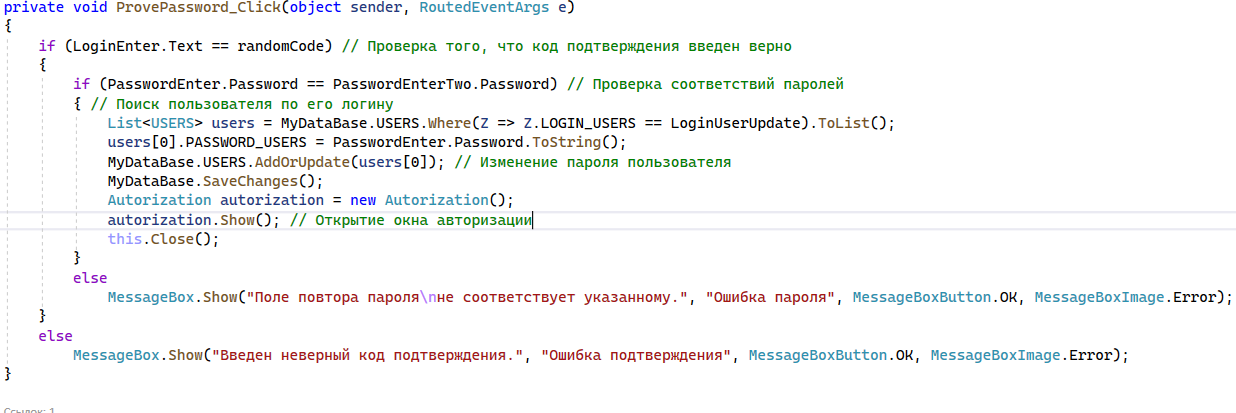


Рисунок 42 – фрагмент кода окна восстановления пароля

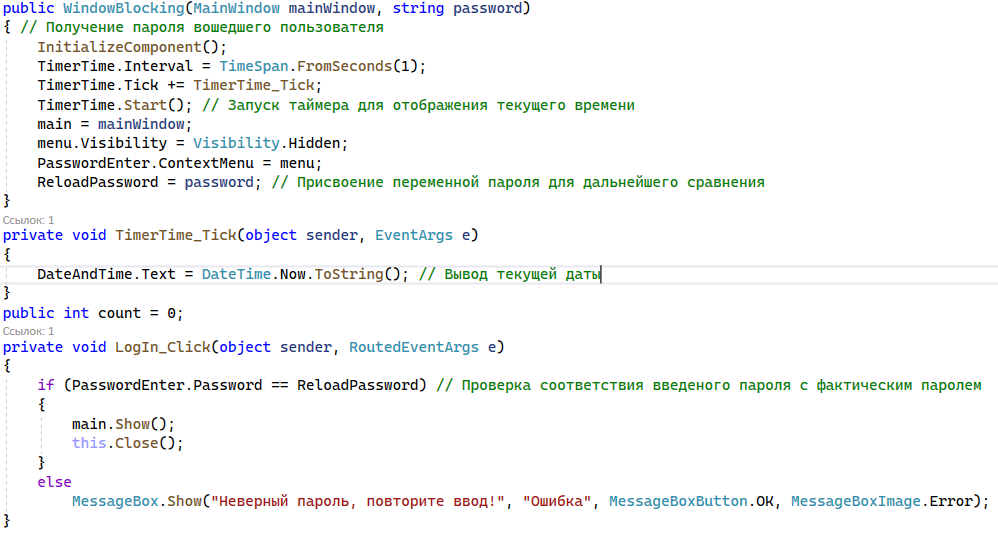


Рисунок 43 – фрагмент кода окна блокировки системы

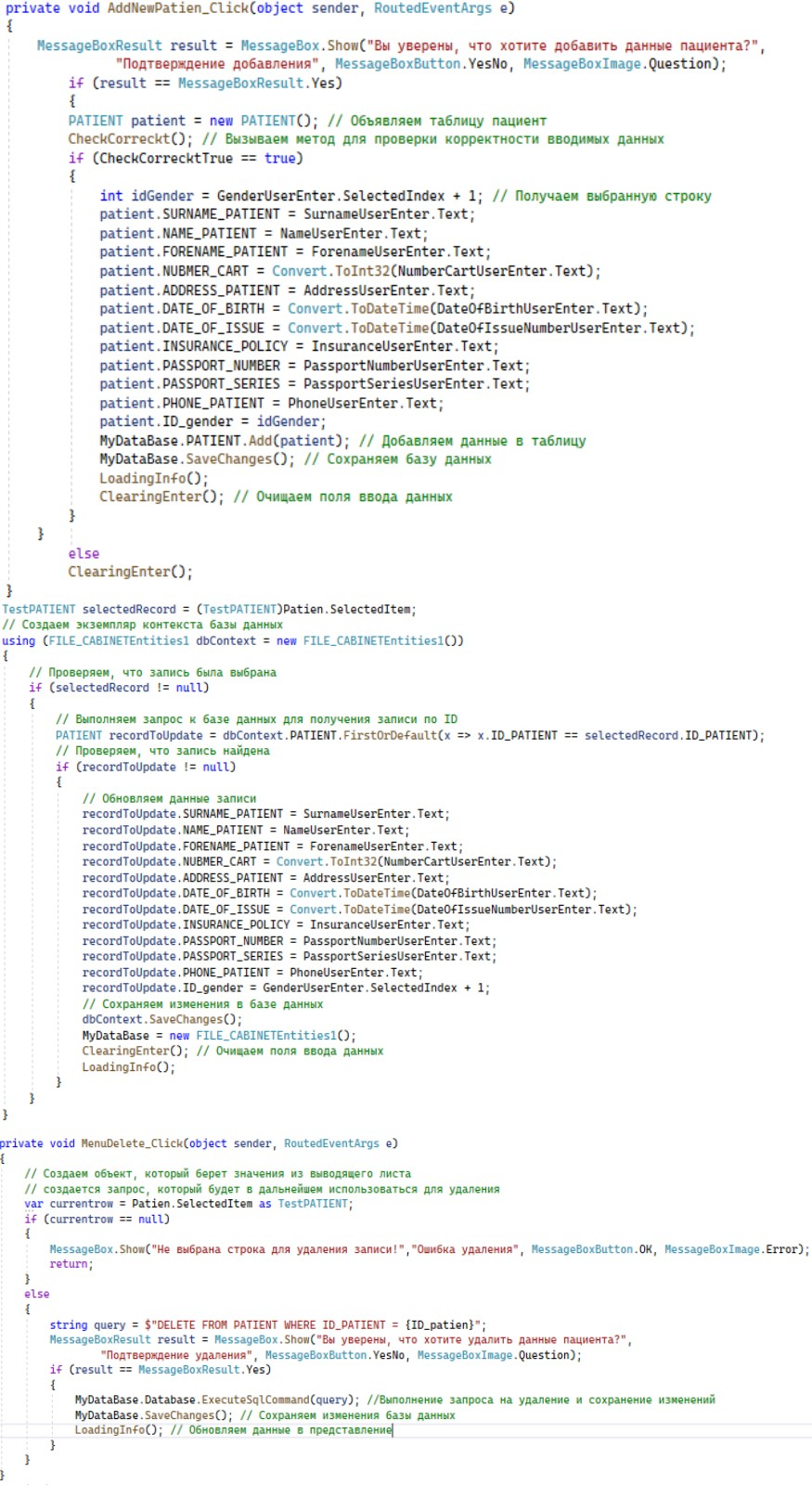


Рисунок 44 – фрагмент кода по добавлению, изменению и удалению данных пациента

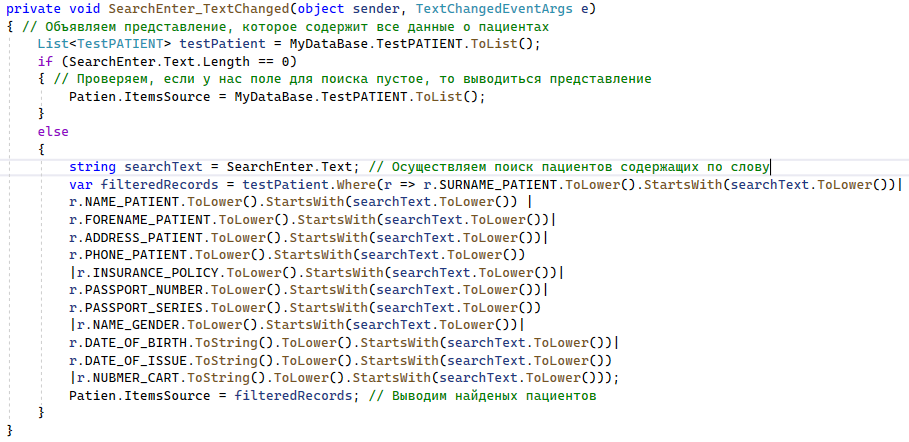


Рисунок 45 – фрагмент кода поиска



Рисунок 46 – фрагмент кода изменение пути сохранения файла осмотра

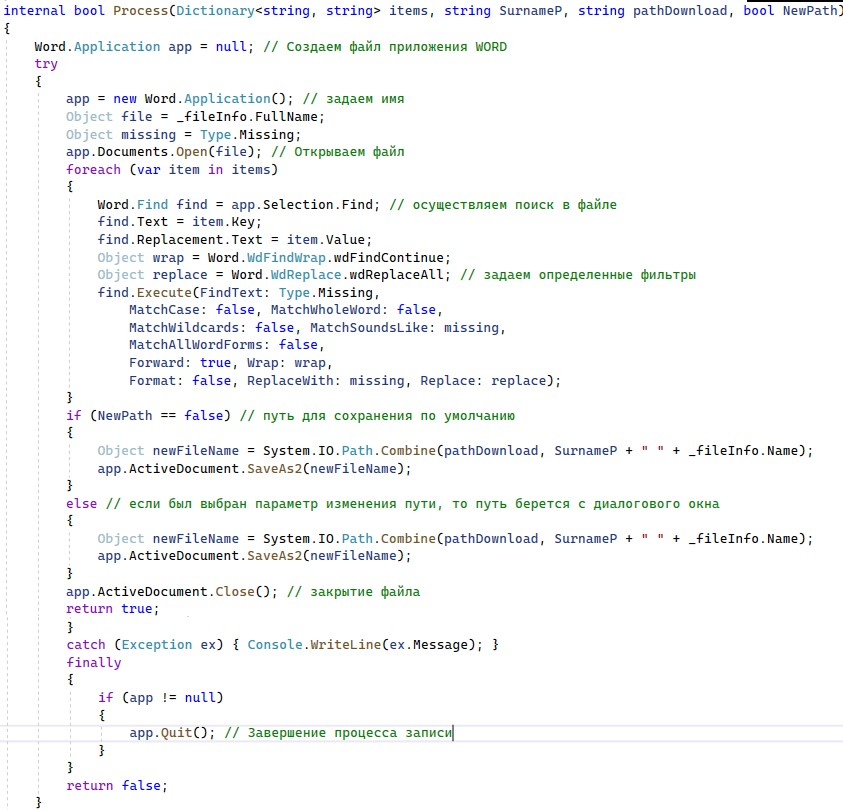


Рисунок 47 – фрагмент кода формирования электронного осмотра

**Заключение**

В заключении необходимо описать:

- какая была достигнута цель;

- какие были выполнены задачи;

- что стало итогом курсового проекта.

**Библиография**

1. Барышникова, Н.А. Экономика организации: учебное пособие для СПО / Н.А. Барышникова, Т.А.Матеуш, М. Г. Миронов - 2-е изд., перераб. и доп. — Москва: Издательство Юрайт, 2018. – 191 с.– ISBN 978-5-534-02672-6.
2. Гаврилов, М.В. Информатика и информационные технологии: учебник/ М.В. Гаврилов, В.А. Климов. – Люберцы: Юрайт, 2019 г. – 383 c.: ил.– ISBN 978-5-534-03051-8.
3. Гагарина, Л.Г. Информационные технологии: Учебное пособие / Л.Г. Гагарина, Я.О. Теплова, Е.Л. Румянцева и др. - М.: Форум, 2018. - 144 c.
4. Гохберг, Г.С. Информационные технологии: Учебник / Г.С. Гохберг. - М.: Academia, 2018. - 474 c.
5. Грибов, В.Д. Экономика организации (предприятия): учебник для СПО. / В.Д. Грибов, В.П. Грузинов, В.А. Кузьменко. – М.: КНОРУС, 2019.– 407 с. – ISBN 978-5-406-05026-2.
6. Грофф, Д.Р. SQL: полное руководство: учебное пособие / Д.Р. Грофф, П.Н. Вайнберг, Э.Д. Оппелъ - Пер. с англ. - М.: ООО "И.Д. Вильяме", 2019. - 960 с.: ил. - ISBN 978-5-8459-1654-9.
7. Дружинина, И.В. Информационные технологии в профессиональной деятельности средних медицинских работников: Учебное пособие / И.В. Дружинина. - СПб.: Лань, 2018. - 112 c.
8. Зубков А. Е. Информатика и языки программирования : практикум / А. Е. Зубков, Е. В. Зубкова, Т. В. Кортева. - Екатеринбург : Изд-во УрГЭУ, 2020. - 98 с.
9. Кислицын Е. В. Современные технологии разработки программного обеспечения : учеб. пособие / Е. В. Кислицын, М. А. Панов. - Екатеринбург : Изд-во УрГЭУ, 2021. - 176 с.
10. Клеппман, М. Высоконагруженные приложения. Программирование, масштабирование, поддержка: учебное пособие / М. Клеппман – СПб.: Питер, 2018. - 640 с.: ил. – ISBN 978-5-4461-0512-0.
11. Коканова, Р.А. Компьютерные информационные технологии в документационном обеспечении управления. Учебное пособие / Р.А. Коканова, А.Ф. Климова. - М.: КноРус, 2018. - 288 c.
12. Коломейченко, А.С. Информационные технологии: Учебное пособие / А.С. Коломейченко, Н.В. Польшакова, О.В. Чеха. - СПб.: Лань, 2018. - 228 c.
13. Кэмпбелл, Л. Базы данных. Инжиниринг надежности: научная литература / Л. Кэмпбелл, Ч. Мейджорс– СПб.: Питер, 2020. – 304 с.: ил. – ISBN 978-5-4461-1310-1.
14. Логинов, В.Н. Информационные технологии управления: Учебное пособие / В.Н. Логинов. - М.: КноРус, 2018. - 416 c.
15. Мельников, В.П. Информационные технологии: Учебник / В.П. Мельников. - М.: Академия, 2018. - 176 c.
16. Михеева, Е.В. Информационные технологии в профессиональной деятельности: учебное пособие / Е.В. Михеева. - М.: Academia, 2019. - 61 c.
17. Плещев В. В. Основы программирования на языках C++ и C# с примерами и упражнениями : учеб. пособие / В. В. Плещев, Е. И. Шишков. - Екатеринбург : Изд-во УрГЭУ, 2018. - 286 с.
18. Управление ИТ-разработкой и внедрением : учеб. пособие / В. П. Часовских, М. П. Воронов, В. Г. Лабунец, Е. Н. Стариков. - Екатеринбург : Изд-во УрГЭУ, 2021. - 173 с.
19. Фуфаев, Д.Э. Разработка и эксплуатация автоматизированных информационных систем: учебник для студ. сред. проф. образования / Д.Э. Фуфаев, Э.В. Фуфаев. – М.: Издательство «Академия», 2018. – 304 с.: ил. - ISBN 978-5-4468-6739-4.
20. Чиркина Н. Г. Информационные системы и технологии : учеб. пособие / Н. Г. Чиркина, М. А. Чиркин . - Екатеринбург : Изд-во УрГЭУ, 2018. - 146 с.
21. Шварц, Б.C. MySQL по максимуму. / П. Зайцев, В. Ткаченко – 3-е изд. – СПб.: Питер, 2018. – 864 с.: ил. – ISBN 978-5-4461-0696-7.
22. Шмелева, А.Г. Информатика. Информационные технологии в профессиональной деятельности: Microsoft Word. Microsoft Excel: теория и применение для решения профессиональных задач / А.Г. Шмелева, А.И. Ладынин. - М.: Ленанд, 2018. - 304 c.

**Приложение А**