МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных Технологий

Кафедра Программной инженерии

Специальность 1-40 01 01 Программное обеспечение информационных технологий

Специализация Программирование интернет-приложений

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**КУРСОВОГО ПРОЕКТА:**

по дисциплине «Объектно-ориентированное программирование»

Тема Программное средство «Система медицинской карты»

Исполнитель

Студент (ка) 2 курса группы 4 Ермаков Кирилл Александрович

(Ф.И.О.)

Руководитель Радиванович Дмитрий Александрович

(учен. степень, звание, должность, подпись, Ф.И.О.)

Курсовой проект защищен с оценкой

Комиссия Радиванович Дмитрий Александрович

(подпись)

Минск 2018

**Реферат**

Пояснительная записка курсового проекта содержит 22 страницы, 15 рисунков и 5 источников литературы.

Темой курсового проекта является разработка Системы медицинской карты.

Пояснительная записка состоит из введения, 4 разделов и заключения.

Во введении представлена необходимость в использовании данной программы.

В первом разделе представлена общая информация о проекте, актуальность и архитектура приложения.

Во втором разделе описана структура проекта и базы данных.

В третьем разделе представлено руководство для пользователя для данного приложения.

В четвёртом разделе представлен процесс тестирования.

**Содержание**

[**Введение** 4](#_Toc514643689)

[**1. Постановка задачи** 5](#_Toc514643690)

[**1.1 Актуальность решаемой задачи** 5](#_Toc514643691)

[**1.2 Средства разработки** 5](#_Toc514643692)

[**2. Разработка архитектуры проекта** 6](#_Toc514643693)

[**2.1 Описание функций, реализуемых программой** 6](#_Toc514643694)

[**2.2 Проектирование классов** 7](#_Toc514643695)

[**2.3 Проектирование базы данных** 10](#_Toc514643696)

[**3. Руководство пользователя** 12](#_Toc514643697)

[**4. Тестирование** 17](#_Toc514643698)

[**Список использованных источников** 20](#_Toc514643699)

[**Приложение А** 21](#_Toc514643700)

**Введение**

Самое ценное у человека – это жизнь, самое ценное в жизни – здоровье. «Здоровье – всему голова», - гласит русская пословица; «Здоровье – самое большое богатство», - вторит ей болгарская мудрость. Отсутствие здоровья, как стесненная в своей свободе жизнь. Соответственно, за здоровьем, как и за любым другим богатством, нужно пристально следить.

Программное средство «Система медицинской карты», о котором и пойдет речь, сможет обеспечить избавление пациента медучреждения от потери и складирования множества бумажек с медицинскими данными, списками лечения, рекомендациями врачей и т.д.

«Система медицинской карты» – это персональная медицинская карта, в которой можно хранить абсолютно все записи о здоровье пациента, не боясь их потерять.

Ежедневно тысячи пациентов обращаются в учреждения здравоохранения. И для того, чтобы ликвидировать бумажную волокиту, очереди, чтобы пациенту больше не было необходимости заказывать свою медицинскую карту в регистратуре к доктору, а сразу, при посещении пациентом кабинета врача, последний мог на месте, зайдя в учетную запись пациента на компьютере, иметь доступ к его медицинской карте, чтобы получить полное представление о состоянии его здоровья. Появился смысл в разработке информатизации этого процесса.

Программное средство «Система медицинской карты» реализует работу с медицинской картой пациента, функциями, которого являются: ведение базы данных пациентов и врачей, просматривание истории поставленных диагнозов/консультаций, возможность производить записи диагнозов/консультаций.

Целью курсового проекта является разработка программного обеспечения для удобства работы с пациентами, связанная с ведением базы данных.

Базой данных называется набор логически связанных данных, который предназначен для удовлетворения информационных потребностей организации/предприятия. Для выполнения задания, поставленного в курсовой работе, необходимо освоить работу по проектированию баз данных и работу с базами данных в выбранной среде программирования.

В качестве интерфейса прикладного программирования был выбран обширный API-интерфейс — Windows PresentationFoundation (WPF), предназначенный для создания настольных программ с графически насыщенным пользовательским интерфейсом.

Для работы с WPF использовался объектно-ориентированный язык программирования с С-подобным синтаксисом — С#, разработанный для создания приложений на платформе Microsoft .NET Framework.

Кроме того, используемая в WPF модель разделения кода и дизайна предполагала работу с декларативным языком описания интерфейса — eXtensibleApplicationMarkupLanguage (XAML).

**1. Постановка задачи**

**1.1 Актуальность решаемой задачи**

В наше время (век современных технологий) появилась необходимость информатизировать бумажные носители на электронные.

Таким образом, данное приложение ещё многие годы будет актуальным, так как, к большому сожалению, люди всю жизнь будут обращаться с недомоганиями и заболеваниями в учреждения здравоохранения для лечения и решения своих проблем, связанных со здоровьем.

**1.2 Средства разработки**

При разработке приложения были использованы:

* интегрированная среда разработки Microsoft Visual Studio 2017;
* программная платформа .NET Framework 5.0;
* язык программирования C#;
* расширяемый язык разметки XAML;
* технология WPF;
* Entity Framework;
* Material Design Library;

*C#* — объектно-ориентированный язык программирования. Разработан в 1998 — 2001 под руководством Андерса Хейлсберга в компании Microsoft как язык разработки приложений для платформы Microsoft .NET Framework.

*XAML* — это декларативный язык разметки. С точки зрения модели программирования .NET Framework язык XAML упрощает создание пользовательского интерфейса для приложения .NET Framework.

# **2. Разработка архитектуры проекта**

## **2.1 Описание функций, реализуемых программой**

Данная программа должна предусматривать функции для реализации процессов работ с графическими примитивами. Варианты использования показаны на диаграмме (рис. 2.1).

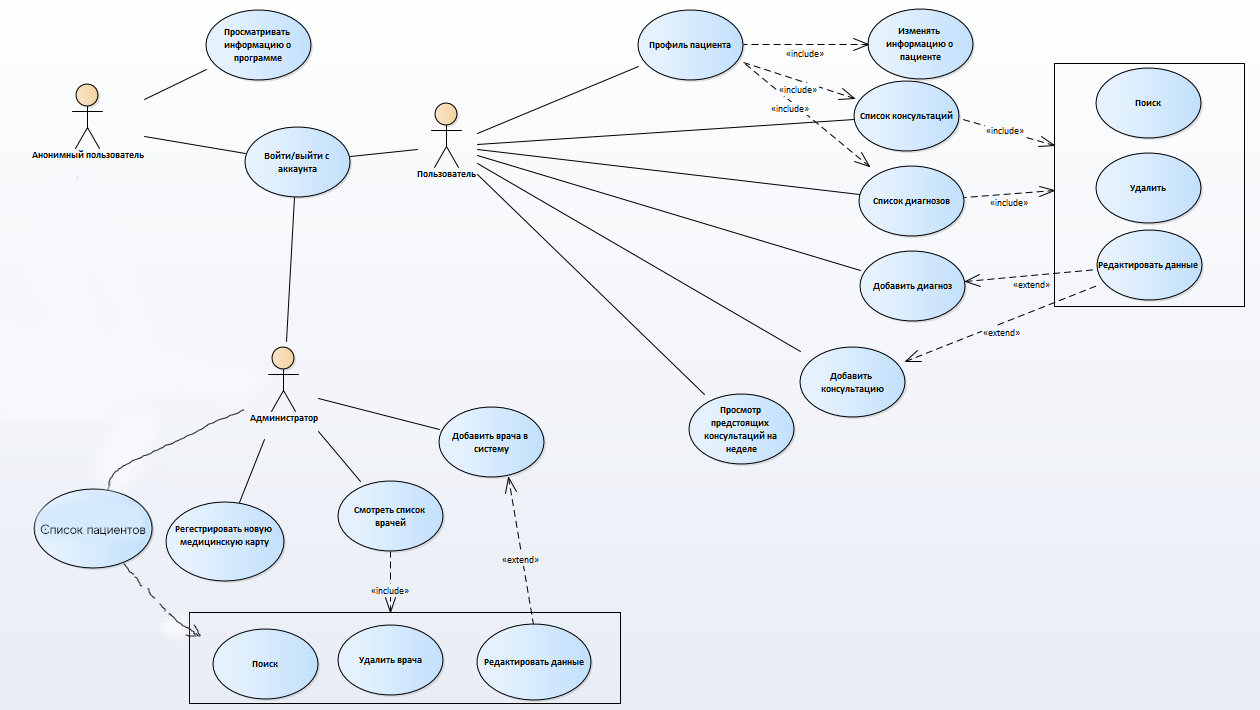


Рисунок 2.1 – Диаграмма вариантов использования

Прецедент «Просматривать информацию о программе» заключается в просмотре информации о программе.

Прецедент «Войти/Выйти с аккаунта» заключается в авторизации (т.е. выбора роли администратора/пользователя) в случае успешного авторизации в учетную запись.

Если мы зашли как «Администратор»:

Прецедент «Смотреть список врачей» заключается в просмотре врачей, находящихся в системе.

Прецедент «Список пациентов» заключается в просмотре пациентов, находящихся в системе.

Прецедент «Регистрировать новую медицинскую карту» заключается добавлении в систему новую медицинскую карту.

Прецедент «Добавить врача в систему» заключается в добавлении в систему нового врача.

Прецедент «Поиск» заключается в поиске данных.

Прецедент «Удалить врача» заключается в удалении врача из системы.

Прецедент «Редактировать данные» заключается в изменении данных.

Если мы зашли как «Пользователь»:

Прецедент «Профиль пациента» заключается в открытии окна в котором находятся все консультации и диагнозы пациента, со возможностью изменения анкетных данных пациента.

Прецедент «Изменить информацию о пациенте» заключается в изменении анкетных данных.

Прецедент «Список консультаций» заключается в просмотре консультаций пациента.

Прецедент «Список диагнозов» заключается в просмотре диагнозов пациента.

Прецедент «Добавить диагноз» заключается в добавлении в систему диагноз пациента.

Прецедент «Добавить консультацию» заключается в добавлении в систему консультацию пациента.

Прецедент «Просмотр предстоящих консультаций на неделе» заключается в просмотре списка консультаций, планируемых на неделе.

Прецедент «Поиск» заключается в поиске данных диагноз/консультация.

Прецедент «Удалить» заключается в удалении данных диагноз/консультация.

Прецедент «Редактировать данные» заключается в редактировании данных диагноз/консультация.

## **2.2 Проектирование классов**

Диаграмма классов приложения представлена в **приложении А**.

*Контейнер сущностей­ –*  представляет собой логическое группирование [наборов сущностей](https://docs.microsoft.com/ru-ru/dotnet/framework/data/adonet/entity-set), [наборов ассоциаций](https://docs.microsoft.com/ru-ru/dotnet/framework/data/adonet/association-set), и [функции imports](https://docs.microsoft.com/ru-ru/dotnet/framework/data/adonet/model-declared-function).

Для контейнера сущностей, определенного в концептуальной модели, должны выполняться следующие условия.

* В каждой концептуальной модели должен быть определен по крайней мере один контейнер сущностей.
* Контейнер сущностей должен иметь уникальное имя внутри концептуальной модели.

Контейнер сущностей может определять наборы сущностей или наборы ассоциаций, которые используют типы сущностей или ассоциации, определенные в одном или нескольких пространствах имен.

В связи с этим были созданы такие классы как: ConsultationDataAccess, DiagnosisDataAccess, DoctorDataAccess, PatientsDataAccess, UsersDataAccess которые хранят методы логического группирования сущностей как получения, добавления, обновления, удаления данных. Дополнительно UsersDataAccess производит валидность(сравнения) пароля и имени пользователя в системе.

ConsultationsPresenter – предназначен для запроса с базы данных консультаций пациента.

DiagnosesPresenter – предназначен для запроса с базы данных диагнозов пациента.

DoctorsPresenter – предназначен для запроса с базы данных врачей.

EditConsultationPresenter – предназначен для заполнения данных консультаций в базу данных и проверки корректных данных, которые добавляются в базу данных.

EditDiagnosisPresenter – предназначен для заполнения данных диагнозов в базу данных и проверки корректных данных, которые добавляются в базу данных.

EditDoctorPresenter – предназначен для заполнения данных врачей в базу данных и проверки корректных данных, которые добавляются в базу данных.

EditPatientPresenter – предназначен для заполнения данных пациентов в базу данных и проверки корректных данных, которые добавляются в базу данных.

EditUserPresenter – предназначен для заполнения данных пользователей в базу данных и проверки корректных данных, которые добавляются в базу данных.

Membership – осуществляет проверку для входа в систему.

PatientsPresenter – предназначен для запроса с базы данных пациентов находящихся в системе.

IConsultationsView, IDiagnosesView, IDoctorsView, IEditConsultationView, IEditDiagnosisView, IEditDoctorView, IEditPatientView, IEditUserView, IPatientsView – это интерфейсы которые реализуют свойства(представляется в виде одного или двух блоков кода для методов доступа [get](https://docs.microsoft.com/ru-ru/dotnet/csharp/language-reference/keywords/get) и (или) [set](https://docs.microsoft.com/ru-ru/dotnet/csharp/language-reference/keywords/set). Блок кода для метода доступа getвыполняется только при считывании свойства, а для метода set – при присвоении свойству нового значения. Свойство без метода доступа set доступно только для чтения. Свойство без метода доступа get – доступно только для записи.).

Также в проекте есть 17 классов окон. Классы окон представлены на рисунке 2.2.

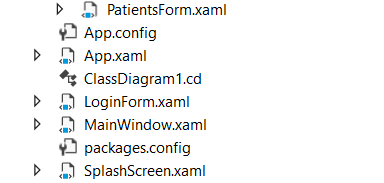


Рисунок. 2.2 – Классы окон

Окно MainWindow в зависимости от роли пользователя: анонимный пользователь содержит Button, для перехода в авторизацию, для просмотра информации о приложении, где задействуются UserControl из папки TranstionsDemo. Пользователь содержит кнопки для перемещения в карту пациента, добавления диагноза, назначения консультаций. StatusBarItem для просмотра в чью медицинскую карту пользователя вошли, а так же для просмотра предстоящих консультаций пациента. Администратор содержит кнопки для перехода в списки врачей/пациентов и добавления в систему врачей/медицинских карт пациентов.

Окна ConsultationsForm, DiagnosesForm, DoctorsForm и PatinetsForm служат для поиска консультаций в назначенный промежуток времени/поиска диагноза/поиска доктора по имени/поиска пациента по имени и номеру, так же четыре окна содержат добавление/редактирование/удаление информации.

Окна EditConsultationForm и EditDiagnosisForm служат для заполнения информацией диагнозов/консультаций, так же в обоих окнах предусмотрен выбор врача и пациента.

Окна EditDoctorForm, EditUserForm предназначены для заполнения/добавления данных пользователя/доктора в систему

Окно EditPatientForm служит для заполнения/исправления анкетных данных о пациентах, содержит DataGrid в котором находятся все консультации/диагнозы с возможностью добавления/удаления/изменения консультаций или диагнозов.

Окно LoginForm служит для авторизации пользователя.

Окно SplashScreen служит для отображения заставки при запуске.

Окна Slide1\_Intro, Slide2\_Intro, Slide3\_Intro, Slide4\_CombineTransitions, Slide7\_MVVM предназначены для отображения информации о программе.

## **2.3 Проектирование базы данных**

Диаграмма полученной базы данных представлена на рисунке 2.3.

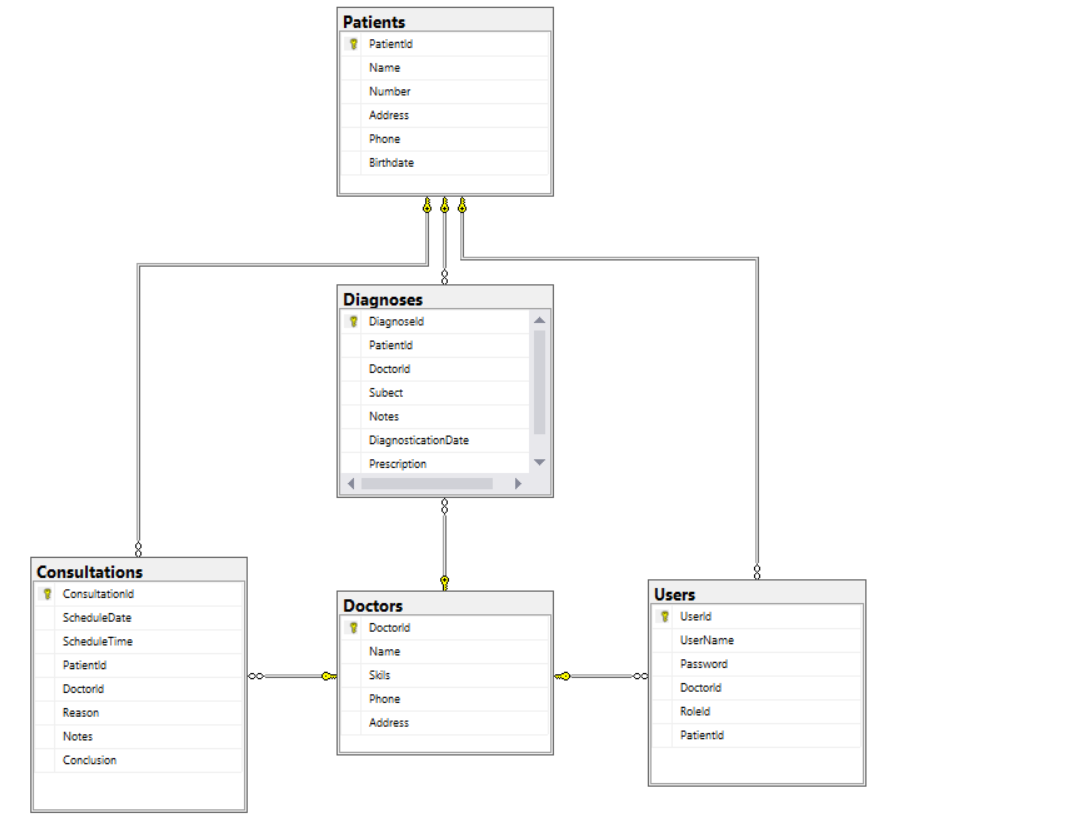


Рис. 2.3 – Диаграмма базы данных

Всего в базе данных 5 таблицы. В таблице Users хранятся все пользователи, зарегистрированные в программе, Patients – информация о пациентах, Doctors – информация о врачах, Consultations – хранит консультации, а также таблица Diagnoses хранит диагнозы.

Таблица Users состоит из шести столбцов:

* UserId– хранит идентификатор пользователя;
* UserName – хранит имя пользователя;
* Password – хранит пароль пользователя;
* DoctorId – хранит идентификатор доктора;
* RoleId – хранит роль пользователя в системе;
* PatientId – хранит идентификатор пациента;

Таблица Patients состоит из шести столбцов:

* PatientId - хранит идентификатор пользователя;
* Name – хранит имя пациента;
* Number – хранит номер присвоенный пациенту;
* Address – хранит адрес пациента;
* Phone – хранит номер телефона пациента
* Birthdate – хранит дату рождения пациента

Таблица Doctors состоит из пяти столбцов:

* DoctorId - хранит идентификатор доктора;
* Name– хранит имя доктора;
* Skils- хранит квалификацию(направление в медицине) доктора;
* Phone – хранит номер телефона доктора;
* Address - хранит адрес доктора;

Таблица Consultations состоит из восьми столбцов:

* ConsultationId - хранит идентификатор консультации;
* ScheduleDate – хранит дату консультации;
* ScheduleTime – хранит время консультации;
* PatientId – хранит пациента на данной консультации;
* DoctorId – хранит доктора назначивший консультацию;
* Reason – хранит причину консультации;
* Notes – хранит примечание к консультации;
* Conclusion – хранит заключение консультации;

Таблица Diagnoses состоит из семи столбцов:

* DiagnoseId - хранит идентификатор диагноза;
* PatientId – хранит пациента которому поставлен диагноз;
* DoctorId– хранит доктора который поставил диагноз;
* Subect – хранит название диагноза;
* Notes – хранит примечания к диагнозу;
* DiagnosticationDate – хранит дату когда был поставлен диагноз;
* Prescription - хранит назначенное лечение;

# **3. Руководство пользователя**

После запуска приложения пользователь увидит главное окно. Оно представлено на рисунке 3.1. В данном окне можно увидеть информацию о программе и произвести авторизацию. Если у вас нет доступа к медицинской карте. Обратитесь к администратору чтоб он создал медицинскую карту.

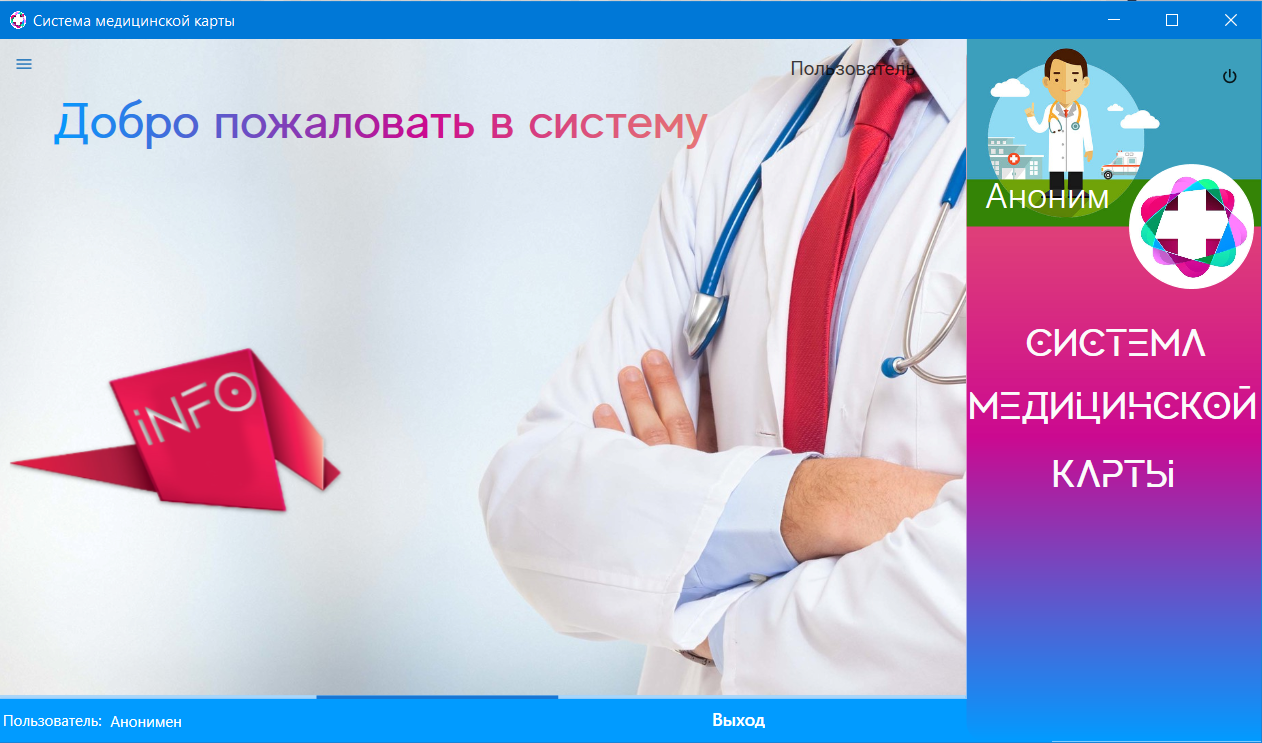


Рисунок 3.1 – Главное окно

Если нажать на кнопку “Войти”, то вы переместитесь в окно авторизации, представлено на рисунке 3.2. После заполнения всех полей нужно нажать кнопку “Войти”, чтобы вы вошли как пользователь или как администратор.

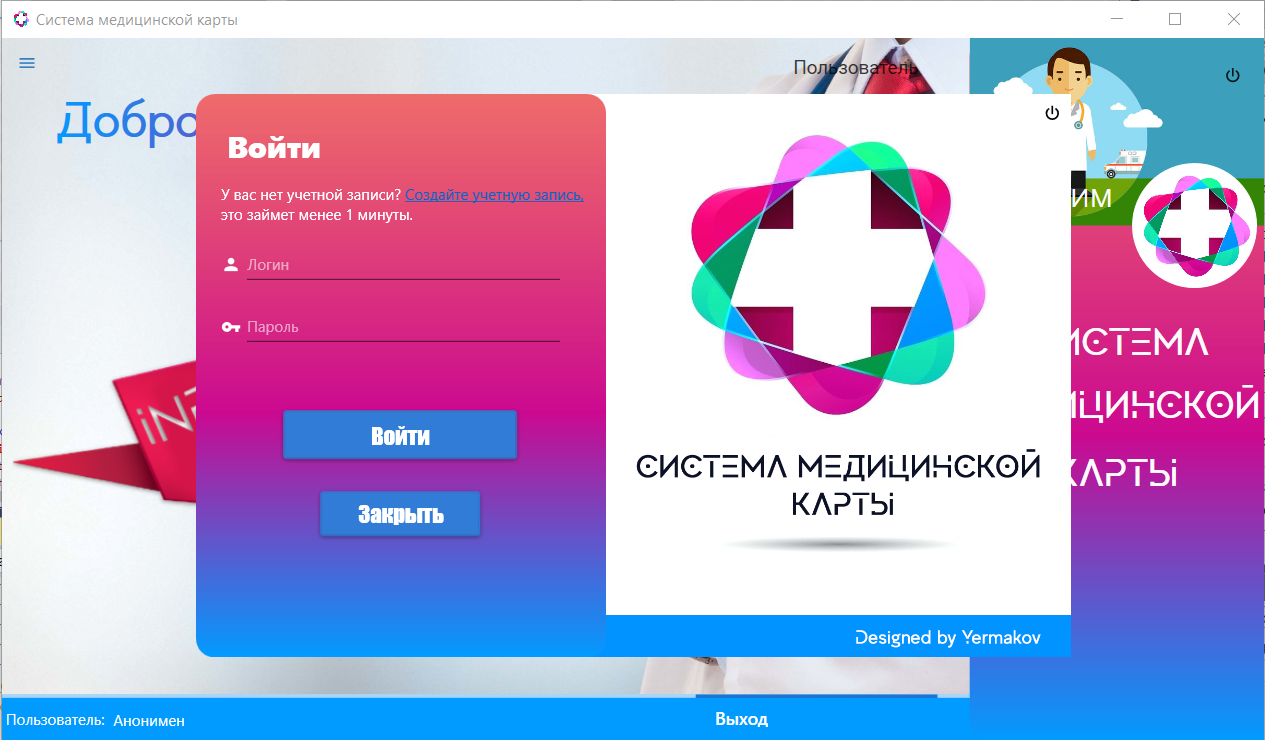


Рисунок 3.2 – Окно авторизации

Если вы войдете в систему медицинской карты(пациента), представлено на рисунке 3.3. То вам будут предложены следующие функции: это войти в профиль пациента, там вы можете редактировать анкетные данные пациента, смотреть диагнозы/консультации с возможностью редактирования, представлено на рисунке 3.4.

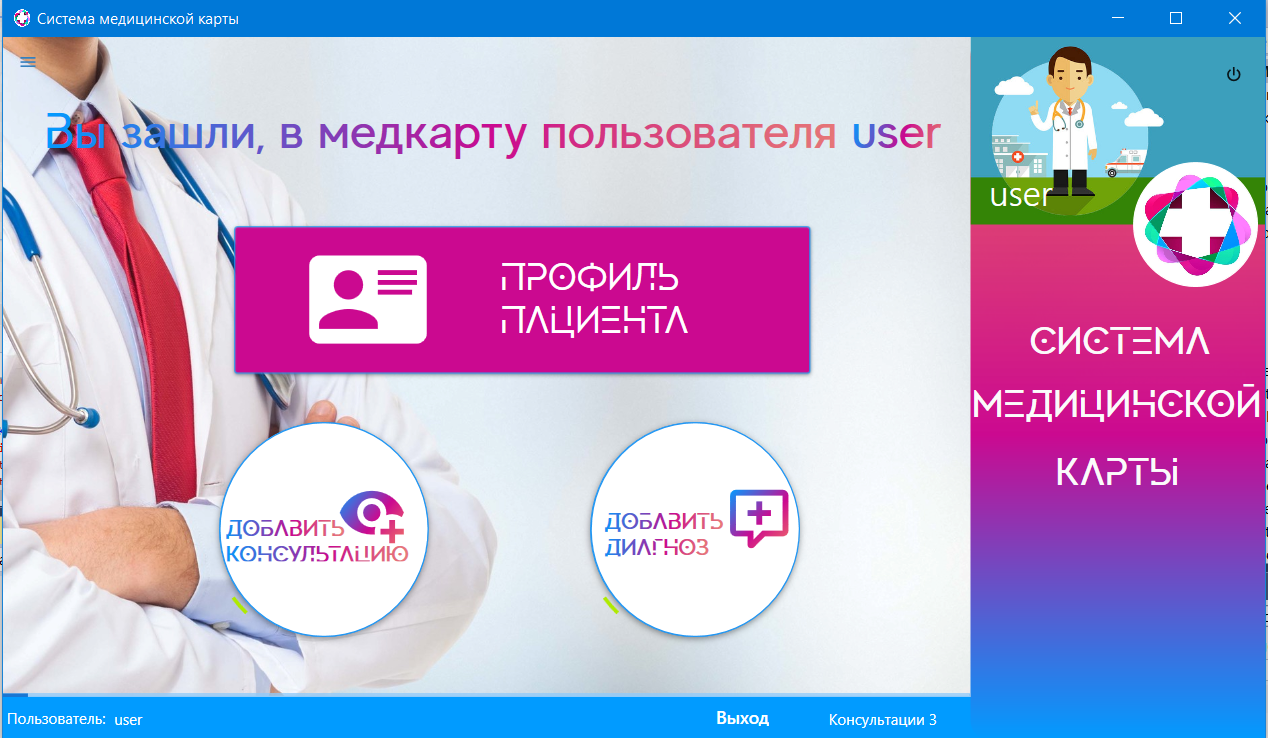
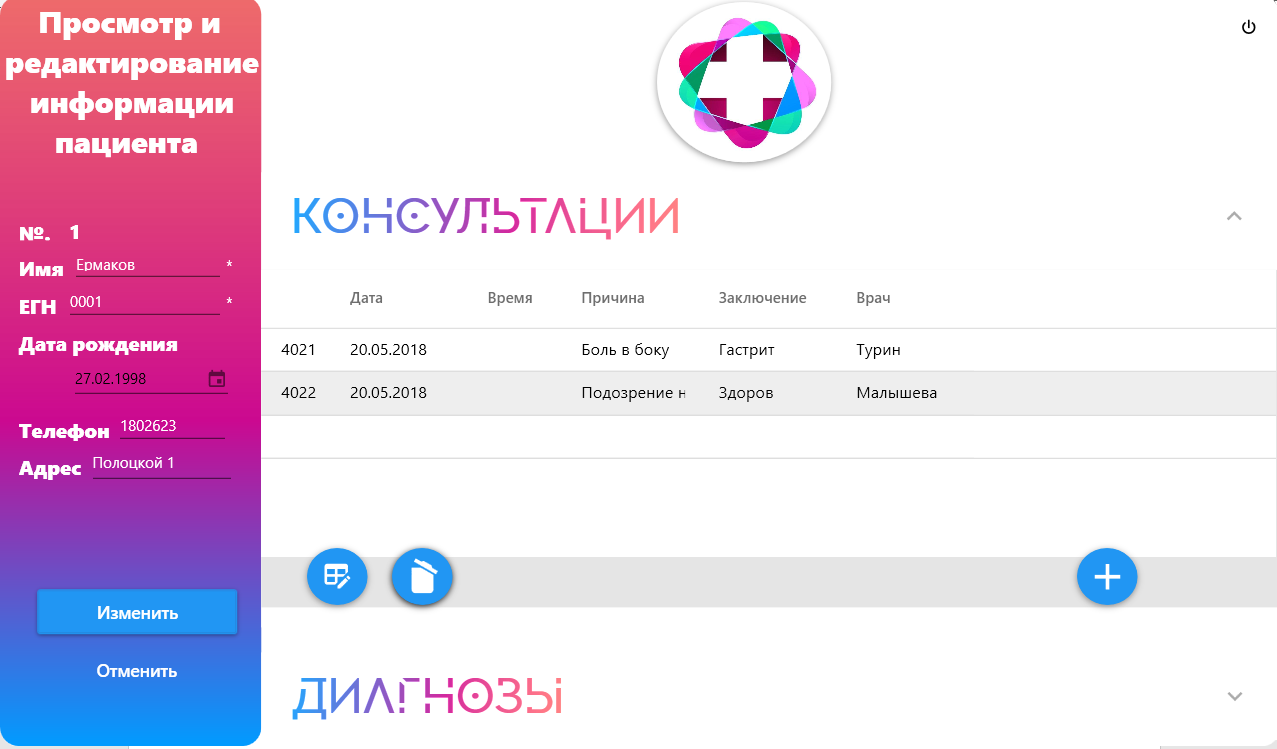


Рисунок 3.3 – Главное окно пользователя

Так же в нижней панели «Главного окна пользователя» отображаются предстоящие на неделе консультации пациента. Если нажать на них, то откроется окно поиска, в котором выведутся консультации.

 Рисунок 3.4 – Профиль пациента

Если вы войдете в «Добавить консультацию» или «Добавить диагноз» (принцип работы аналогичен), представлено на рисунке 3.5. То мы сможем добавить информацию с систему о поставленном диагнозе/консультации с возможностью выбора доктора/пациента.

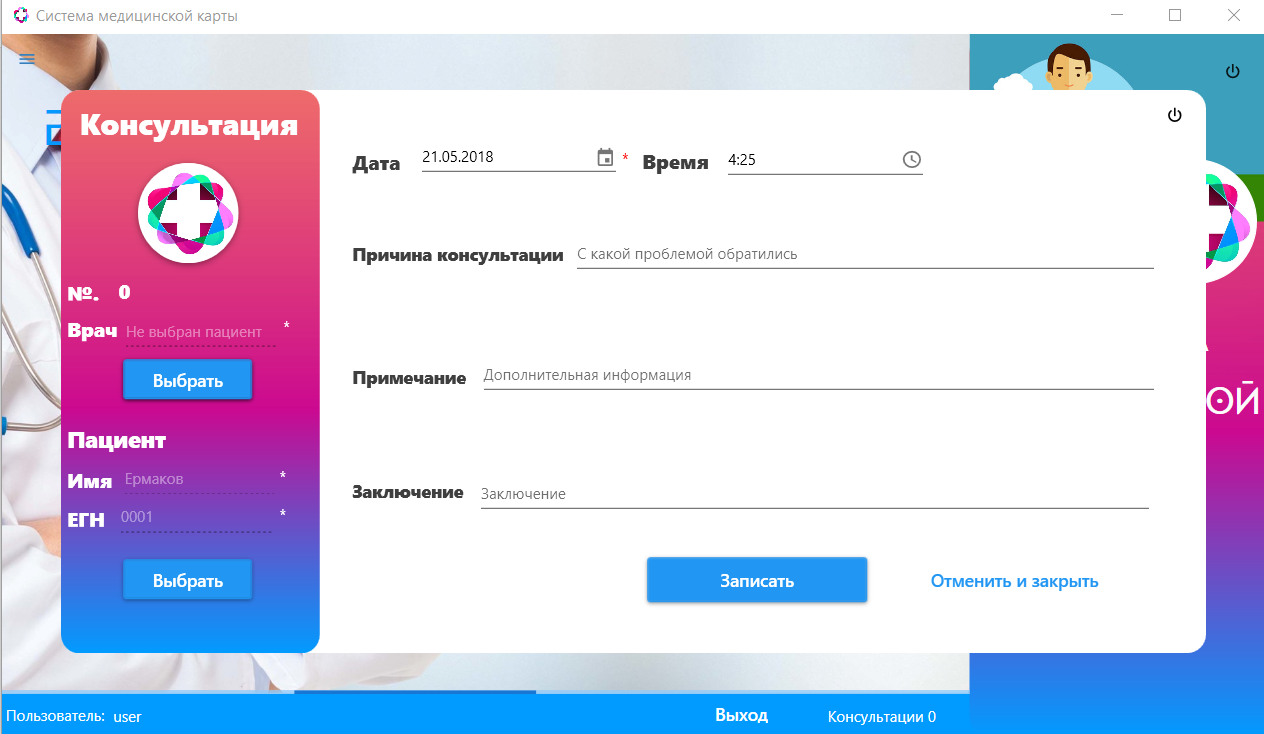


Рисунок 3.5 – Консультация

При нажатии кнопки «Поиск диагнозов» или «Поиск консультаций» (работа аналогична) появится окно, в котором можно произвести поиск, добавить, редактировать, удалить консультацию/диагноз, представлено на рисунке 3.6.

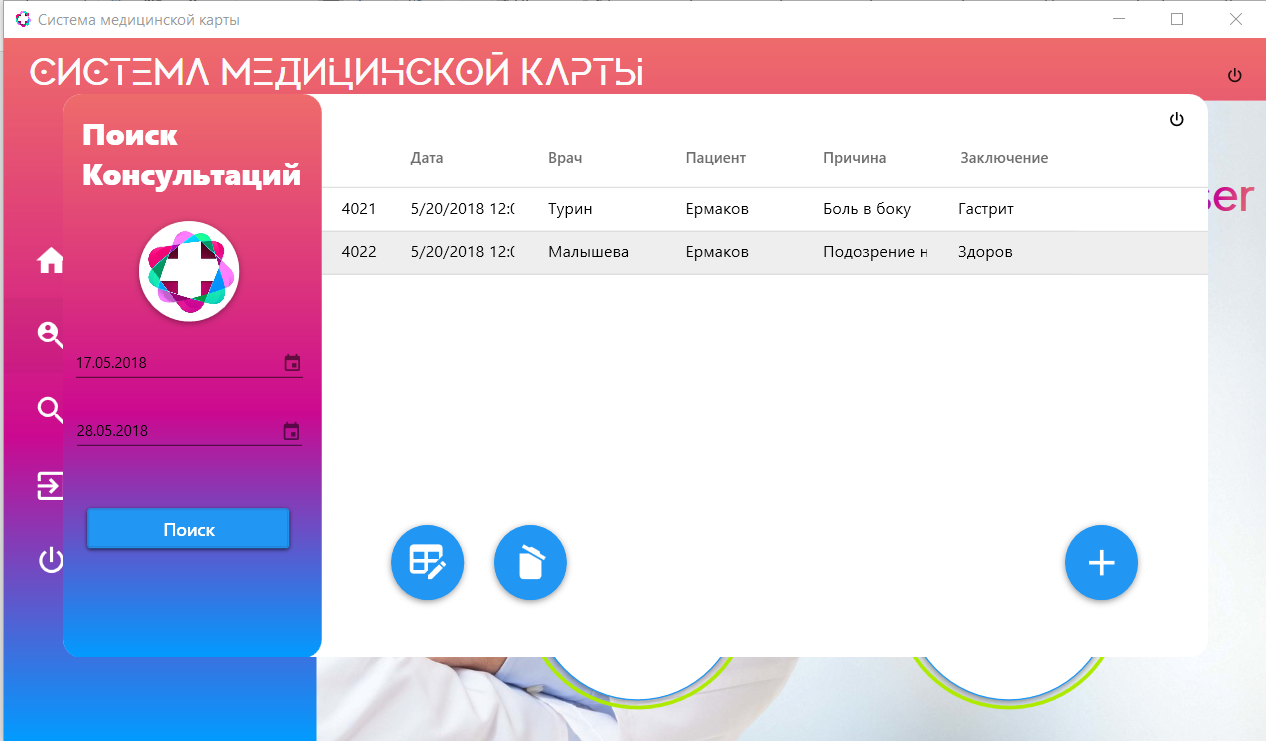


Рисунок 3.6 – Окно поиска консультаций

Если вы войдете как «Администратор», то вам будут доступны следующие функции «Добавление врача», «Новая карточка», «Список врачей», «Список пациентов», представлено на рисунке 3.7.

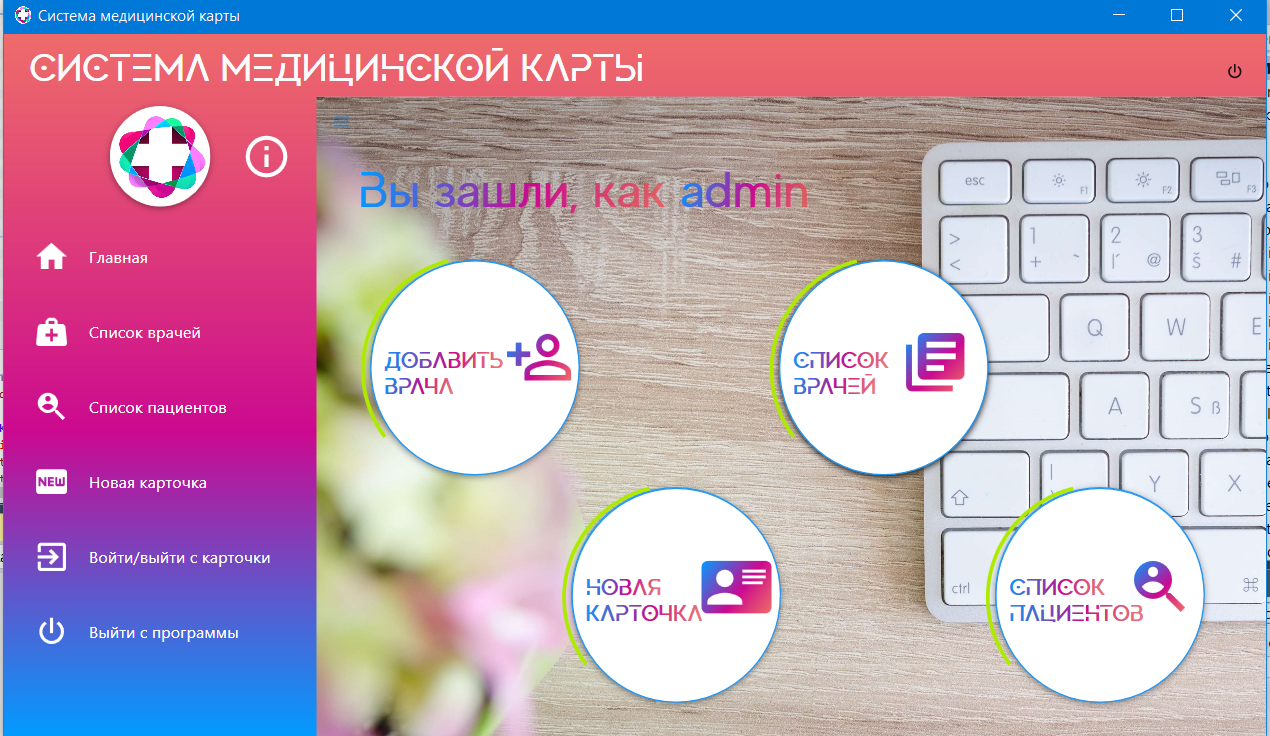


Рисунок 3.7– Окно Администратора

Если войти в «Добавить врача» или «Новая карточка» (работа аналогична), представлено на рисунке 3.8. То здесь вы заполняете корректные данные благодаря которой в системе будет новый пациент/врач.

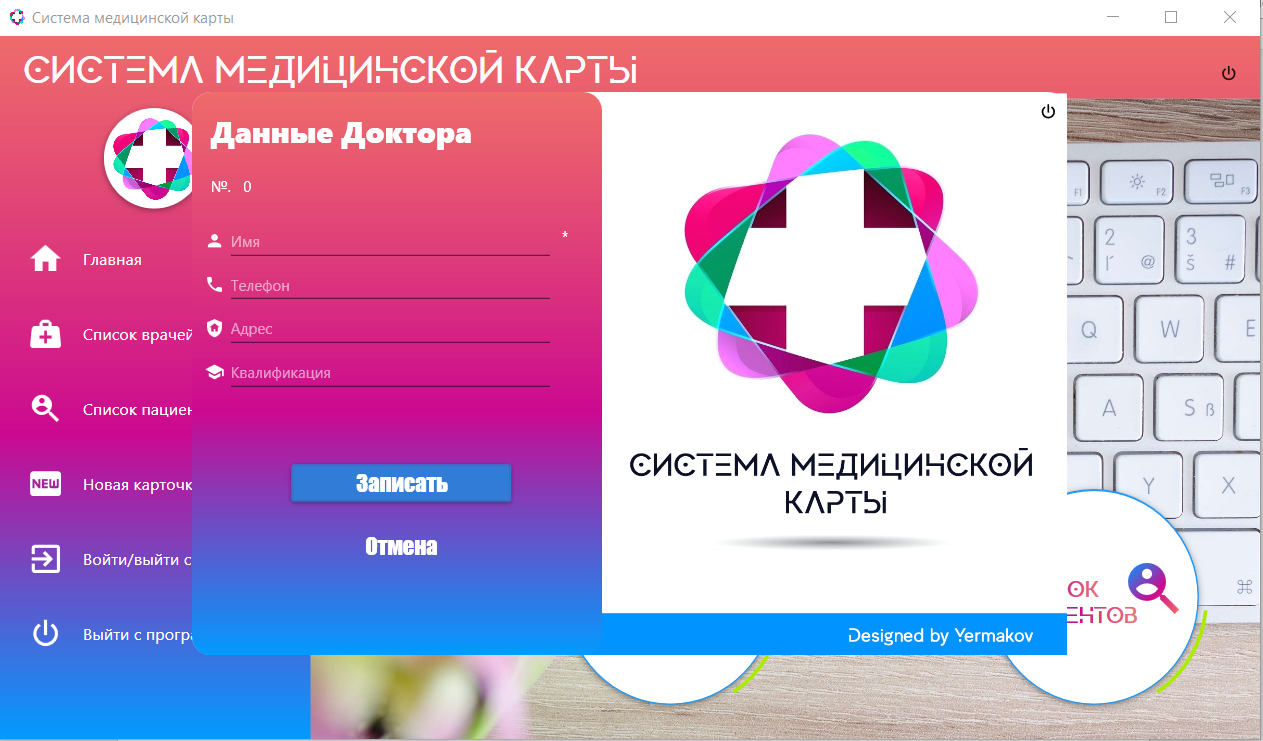


Рисунок 3.8 – Окно добавление доктора в систему

Если войти в «Поиск врачей» или «Поиск пациентов»(работа аналогична), представлено на рисунке 3.9. То вам будет представлена возможность производить поиск, удаление, редактирование и добавление новых данных о врачах.

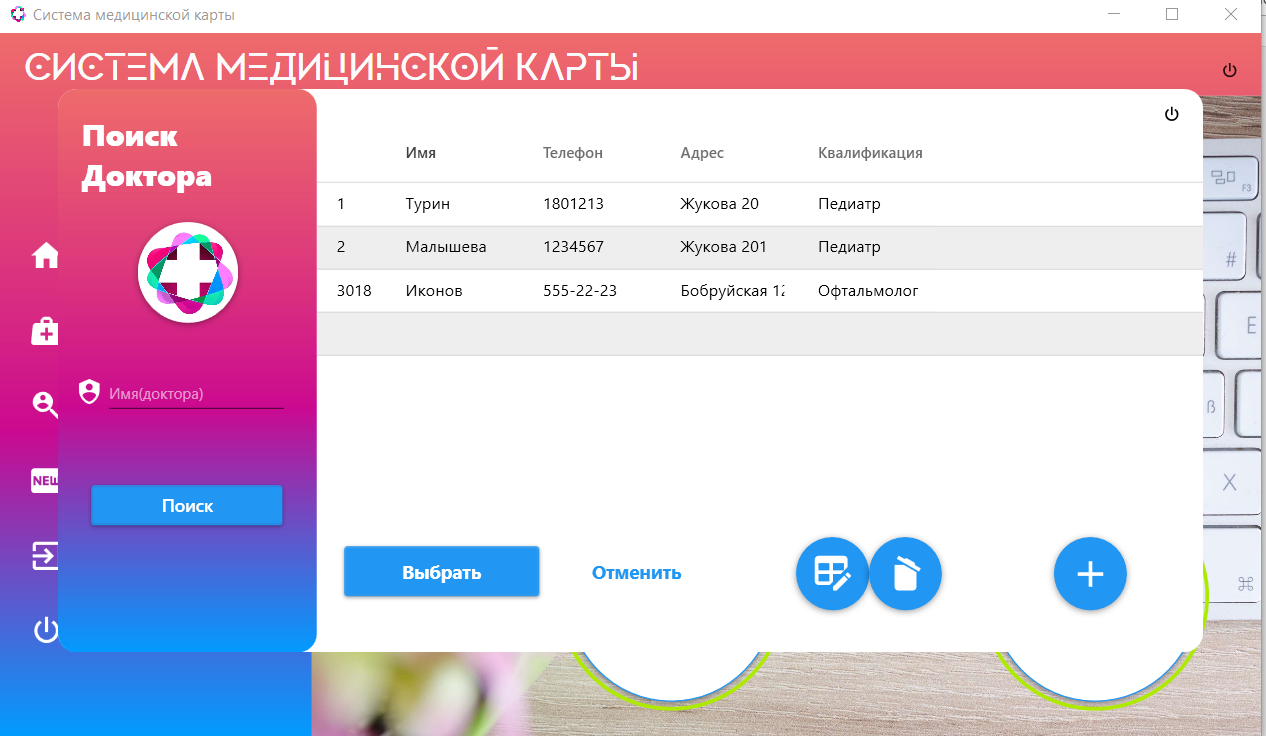


Рисунок 3.9 – Окно поиска доктора

# **4. Тестирование**

При тестировании данного приложения были проведены следующие тесты на ситуации, которые могли бы привести к ошибке:

* тесты авторизации и регистрации;
* тест изменения данных;

В момент регистрации возможна ситуация, когда пользователь вводит недостаточной безопасности пароль. Обработка данного исключения продемонстрирована на рисунке 4.1.

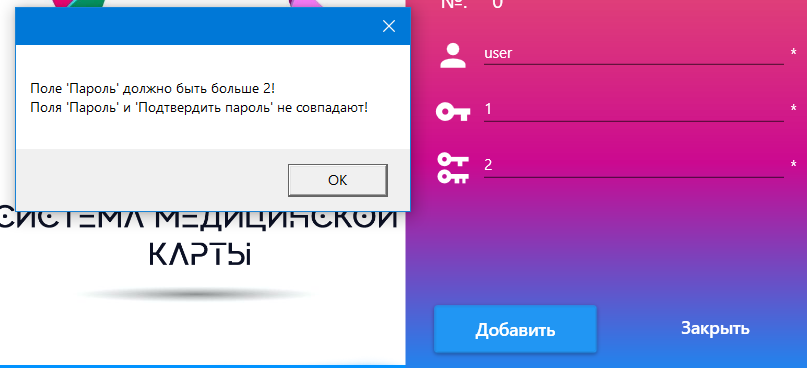


Рисунок 4.1 – Тестирование регистрации

В момент авторизации возможна ситуация, когда пользователь вводит неверные данные пользователя, который ещё не зарегистрирован. Обработка данного исключения продемонстрирована на рисунке 4.2.

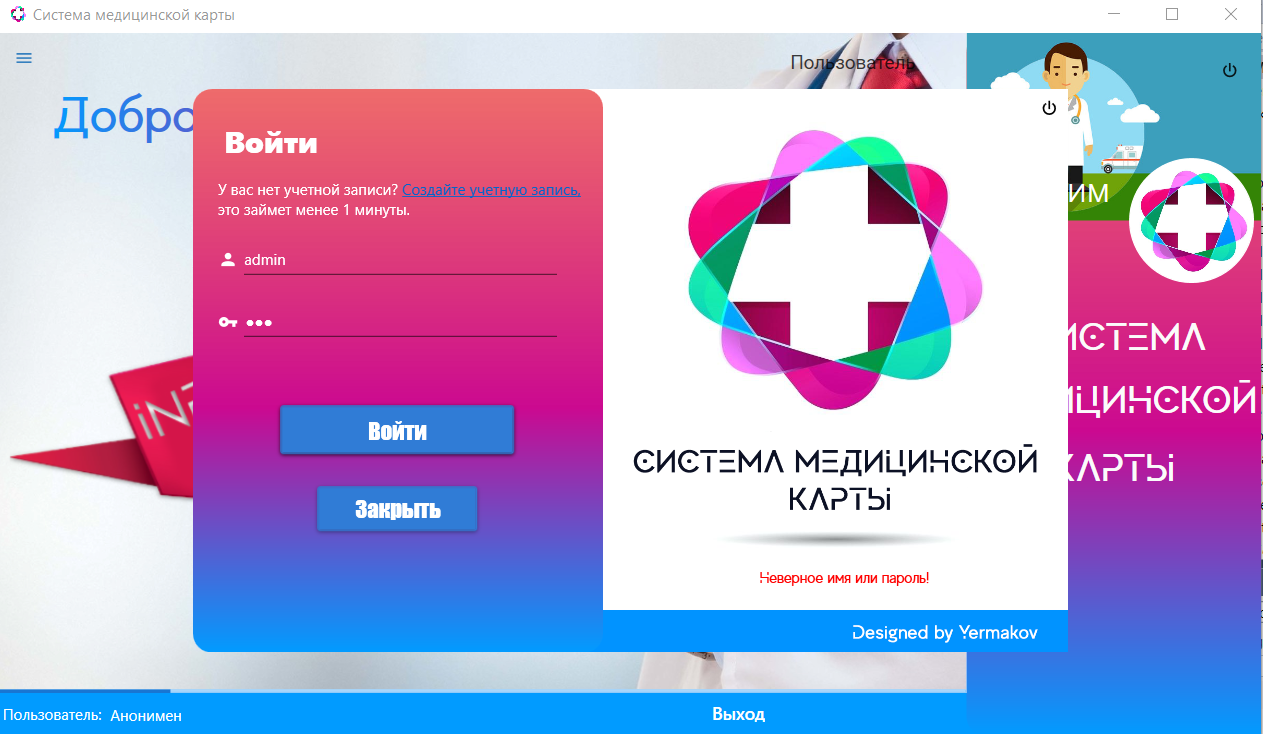


Рисунок 4.2– Тестирование авторизации

Когда мы попытаемся изменить анкетные данные(обнулить поля со звездочкой). Система не допустит чтоб пациент был не идентифицирован Обработка данного исключения продемонстрирована на рисунке 4.3.

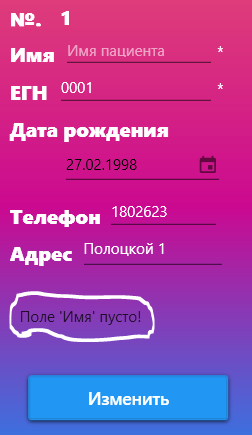


Рисунок 4.3 – Тестирование сохранения одинаковых приоритетов

**Заключение**

В процессе решения поставленной задачи была достигнута поставленная цель по созданию программного средства «Система медицинской карты». При разработке были выполнены все пункты из указанного списка предполагаемого основного функционала приложения.

В программном средстве были реализованы следующие функции:

* Анонимный пользователь:
  + Выполнять авторизацию;
  + Просмотр информацию о приложении;
* Администратор:
  + Добавить в систему врача/карту пациента;
  + Редактировать информацию о враче;
* Удалять врача/пациента из системы;
* Просматривать список врачей/пациентов.
* Пользователь:
* Добавлять информацию о пациенте;
* Добавлять записи осмотров и диагнозов;
* Удалять выбранный осмотр или диагноз;
* Редактировать информацию осмотров и диагнозов;
* Производить поиск, просматривать список осмотров, диагнозов;
* Просмотр консультаций, назначенных на неделю пациенту.

В соответствии с полученным результатом работы программы можно сделать вывод, что разработанная программа работает, верно, а требования технического задания выполнены в полном объеме.

**Список использованных источников**

1. Пацей, Н.В. Курс лекций по языку программирования С# / Н.В. Пацей. – Минск: БГТУ, 2016. – 175 с.
2. Пацей, Н.В. Технология разработки программного обеспечения / Н.В. Пацей. – Минск: БГТУ, 2016. – 129 с.
3. MSDN сеть разработчиков в Microsoft [Электронный ресурс] / Режим доступа: <http://msdn.microsoft.com/library/rus/>. Дата доступа: 20.04.2018
4. METANIT.COM Сайт о программировании [Электронный ресурс] / Режим доступа: <https://metanit.com>. Дата доступа: 20.05.2018
5. ProfessorWeb .NET & Web Programming [Электронный ресурс] / Режим доступа: <https://professorweb.ru> Дата доступа: 13.04.2018

**Приложение А**

