МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
Учреждения образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий
Кафедра Программной инженерии
Специальность 1-40 01 01 Программное обеспечение информационных технологий
Направление специальности 1-40 01 01 10 Программное обеспечение информационных технологий (программирование интернет-изданий)

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА КУРСОВОГО ПРОЕКТА:

по дисциплине «Объектно-ориентированные технологии программирования и стандарты проектирования»

Тема «Программное средство «Терапевт»

Исполнитель
студентка 2 курса группы 5 Дубень Полина Васильевна
(Ф.И.О.)

Руководитель работы Панченко О. Л.
(учен. степень, звание, должность, подпись, Ф.И.О.)

Курсовой проект защищен с оценкой _____

Председатель Пацей Н.В.
(подпись)

Минск 2019

Содержание

ВВЕДЕНИЕ

В современном мире информационные технологии позволяют автоматизировать и упростить работу во многих сферах деятельности человека. Базы данных, быстрый поиск и сортировка, автоматическое вычисление данных позволяют упростить в частности работу терапевта. Это и является целью моего курсового проекта – создание простого и удобного программного средства, позволяющего организовать работу терапевта с использованием компьютера, а не бумажных карт. Это позволит ускорить и упростить работу врача и учреждения здравоохранения в целом, потому что освободит то время, которое тратится на запись информации в карту, выписку справок и рецептов, поиск нужной карты человеком в регистратуре, перенос её в кабинет врача и так далее.

Данное программное средство реализовано на технологии Windows Presentation Foundation, выполнено с использованием ООП, взаимодействует с базой данных. Отображение, бизнес логика и база данных отделены друг от друга в целях простого расширения. Управление программой интуитивно понятно и удобно.

Программное средство в данный момент актуально, многие поликлиники и больницы переходят на использование электронных карт, программ для записей посещений, электронных рецептов.

Для выполнения данного программного средства были поставлены следующие задачи:

1. Обзор литературы по теме, примеры похожих программных средств, анализ их с критической стороны;
2. Установление требований к программному средству, описание функциональности;
3. Проектирование, создание логических схем работы программного средства;
4. Программирование, отладка модулей проекта, сборка и комплексная отладка. Описание классов, структур, атрибутов и методов;
5. Проведение тестирования программного средства;
6. Разработка руководства по установке и использованию приложения.

Решение каждой из задач отображено соответственно в главах:

1. Аналитический обзор литературы;
2. Анализ требований к программному средству и разработка функциональных требований;
3. Проектирование программного средства;
4. Создание (реализация) программного средства;
5. Тестирование, проверка работоспособности и анализ полученных результатов;
6. Руководство по установке и использованию.

1 Аналитический обзор литературы

Для составления технических требований к проекту был проведен анализ похожих программных средств. В процессе анализа были рассмотрены достоинства и недостатки найденных проектов, на основе которых разрабатывались технические требования для разрабатываемого программного средства. Ссылки на материалы, используемые в данной главе, предоставлены в списке литературы.

Приведенная на рисунке 1.1 форма регистрации на приём содержит сразу несколько недостатков.

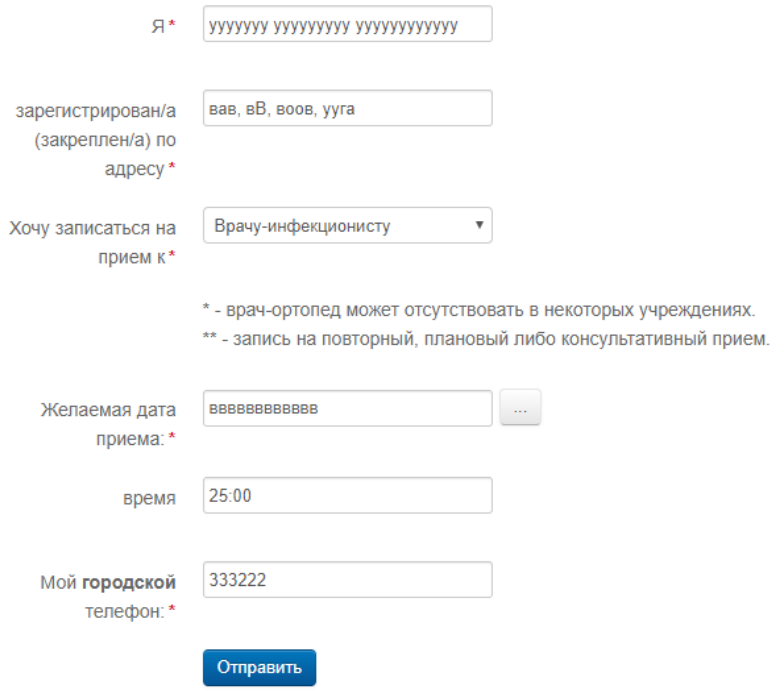


Рисунок 1.1 – Форма регистрации на приём

В полях «Желаемая дата приёма» и «время» могут содержаться недопустимые значения, и при отправке формы не выдаётся ошибок. Это решается валидацией вводимых данных и созданием шаблонов для полей. Также к достоинствам нельзя отнести дизайн: нерационально используется пространство формы (много незадействованного места), стандартный дизайн полей и текста, неинтересная цветовая гамма. Из плюсов можно отметить разделение полей на обязательные и необязательные. Данная форма содержится на сайте действующей поликлиники.

Также подобные формы регистрации можно найти в мобильных приложениях на схожую тематику, например, отраженная на рисунке 1.2 форма содержится в мобильном приложении «Damumed – Управляй своим здоровьем».

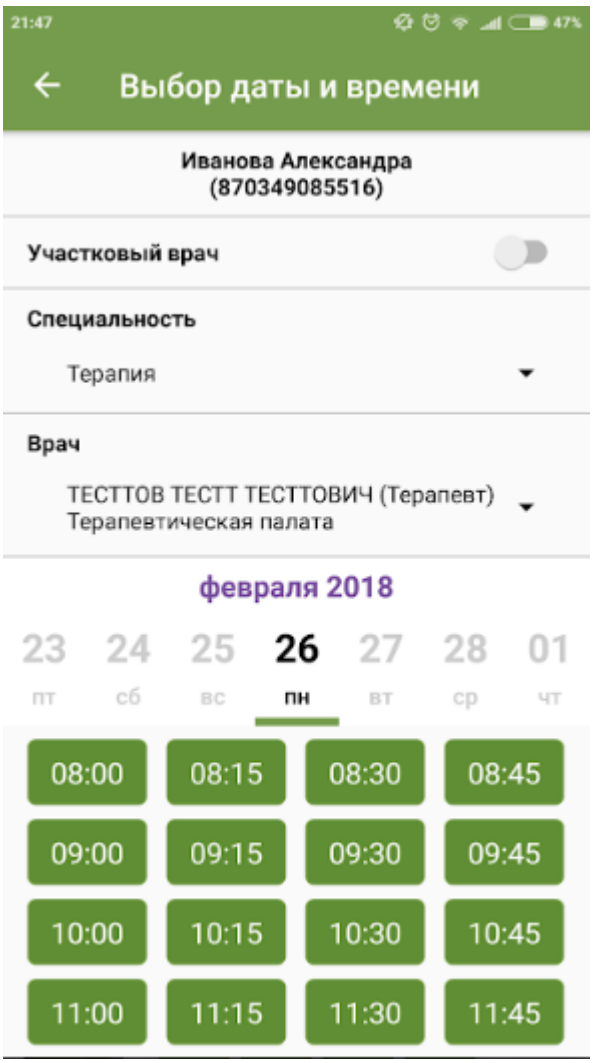


Рисунок 1.2 – Форма регистрации на приём в мобильном приложении

К достоинствам этого приложения можно отнести малое количество полей для заполнения (пользователям удобно заполнять их на экране телефона или планшета), дизайн, а также удобная организация выбора даты и времени приёма. Обнаруженные недостатки:

– заголовок «Выбор даты и времени» не совсем соответствует действительности (на форме также присутствуют поля для выбора специальности и врача);

– интуитивно не понятно предназначение кнопки «Участковый врач»;

– на форме содержится лишняя информация (информация о клиенте).

На рисунке 1.3 отображено программное средство, предназначенное для учёта посещения больных, контроля за приёмом лекарственных препаратов. Данное приложение имеет широкую базу данных, содержит большое количество вариаций таблиц для вывода информации. К достоинствам программы относятся возможность сохранять отчёты, отправлять их на печать, широкий функционал и наличие справочной информации. Однако программа сложна в понимании и освоении, содержит очень широкий список функций, меню с множеством элементов и рабочую область, в которой сложно ориентироваться. Все элементы и

функционал расположен на одном окне, что сильно усложняет работу с программой для неопытного пользователя.

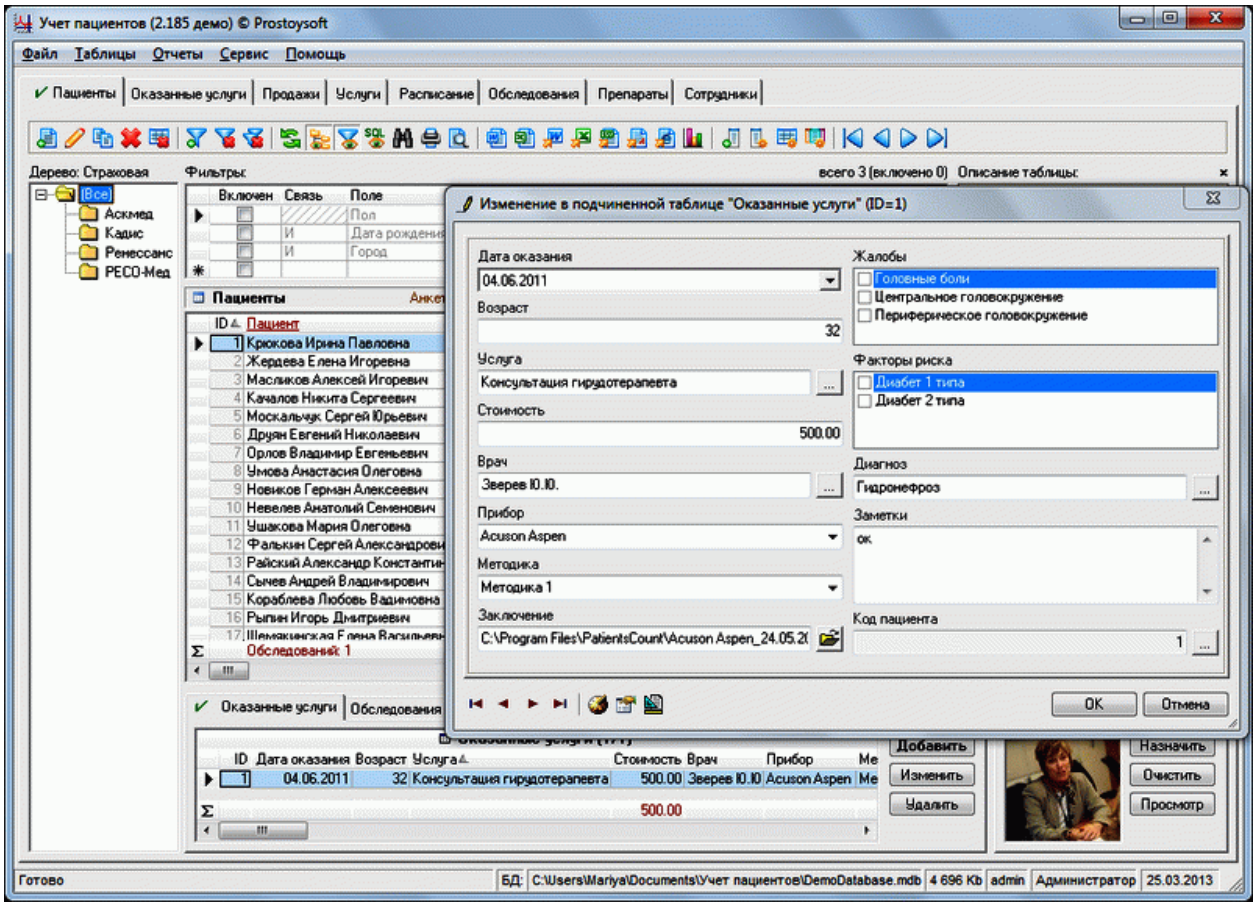


Рисунок 1.3 – Программное средство учёта посещений больных

Программное средство, отображенное на рисунке 1.4, является лучшей программой, которая была найдена в сети, на заданную тему. В ней реализован поиск клиента и посещения по нескольким критериям, присутствует автообновление данных, шаблоны для печати, интуитивно понятный дизайн. Присутствует разделение функционала в зависимости от учётной записи, справочная информация. Реализованы запись в карту пациента, вывод расписания посещений, есть возможность автоматической выписки направления на анализы, широкая база данных.

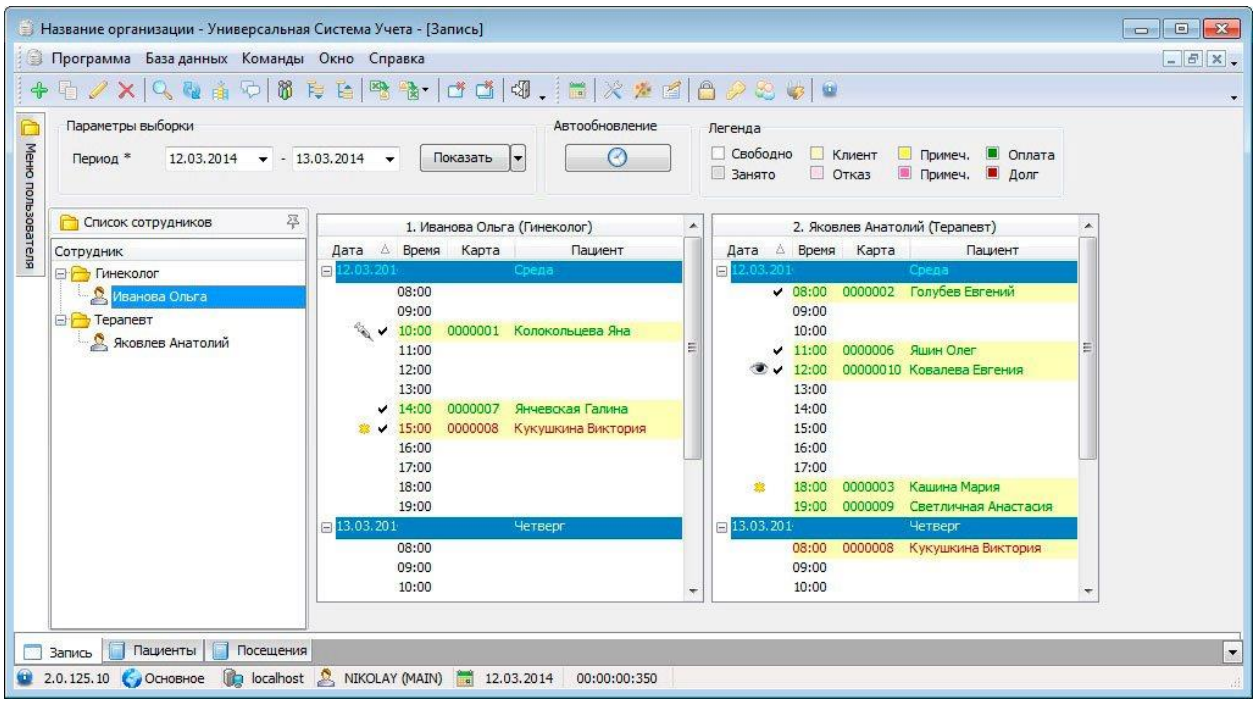


Рисунок 1.4 – Программное средство учёта посещений

На рисунке 1.5 продемонстрировано программное средство для учёта посещений.

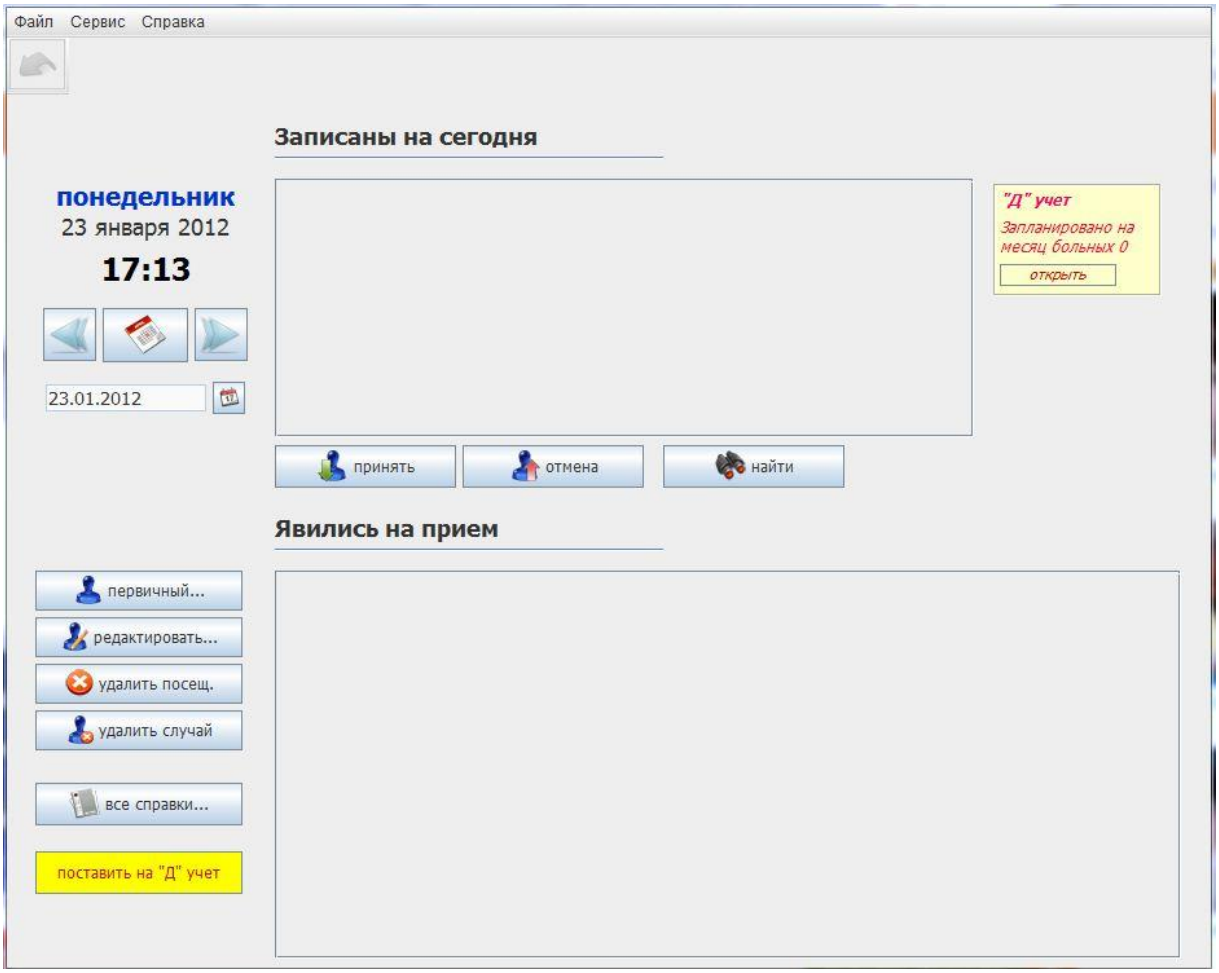


Рисунок 1.5 – Программа амбулаторный приём

Преимущества данной программы: удобный интерфейс, предварительный просмотр документов перед печатью, что позволяет визуально контролировать правильность его заполнения, самодостаточность – отсутствие необходимости использования совместно с другими офисными приложениями, подробная инструкция по работе с программой. Данное программное средство простое в использовании, имеет минимальный функционал. К явным недостаткам можно отнести именование кнопок (использование сокращений, различное выравнивание).

Согласно найденному в интернете источнику (см. список литературы), ПВТ представил программные решения для использования в сфере здравоохранения. Это значит, что разработка приложений на данную тематику актуальна и востребована.

Проанализировав программы на схожую тематику, были разработаны технические требования к разработке проекта. Проект должен:

– содержать валидацию вводимых данных;

– иметь интуитивно понятное и простое управление;

– обращаться к базе данных, успешно извлекать и дополнять данные, осуществлять поиск по базе и автоматическую запись некоторых полей;

– иметь различный функционал в зависимости от учётной записи. Терапевт может вводить информацию о приёме, назначать новые приёмы, изменять информацию в базе данных, выписывать рецепты. Пациент может только записываться на новые приёмы, администратор может добавлять и изменять записи терапевтов;

– выглядеть аккуратно, не иметь лишней информации, адаптировать размеры и расположение элементов управления в зависимости от размера окна.

Эти пункты и составляют техническое задание разрабатываемого проекта.

2 Анализ требований к программному средству и разработка функциональных требований

При запуске приложения открывается окно регистрации. Программное средство «Терапевт» имеет два типа учётных записей: терапевт и пациент. В зависимости от типа записи программное средство будет реализовывать различный функционал. Существует три варианта авторизации: для входа в запись терапевта или администратора необходимо ввести логин и пароль, для входа в запись пациента требуется нажать на кнопку «Записаться на приём».

Учётная запись терапевта будет иметь функционал значительно шире остальных. После окна регистрации для терапевта откроется окно с единственной кнопкой «Начать приём» и меню, функции которого будут рассмотрены далее. При нажатии на кнопку запустится отсчёт времени, который прекратится после завершения приёма, что позволит автоматически записать время, потраченное на отдельного пациента, в базу данных. После начала приёма откроется окно для выбора пациента, с которым будет проводиться приём. В данном окне можно будет организовать поиск пациента в базе по имеющимся данным. После выбора записи из результатов поиска откроется следующее окно. Если необходимо изменить информацию о посетителе, требуется нажать на его записи в списке результатов поиска, а затем нажать на кнопку «Изменить». В таком случае откроется окно с информацией о клиенте, в которой можно будет указать новые значения. Также будет присутствовать возможность добавить нового пациента, если он пришёл на приём впервые. При добавлении нового пациента откроется окно для заполнения информации о нём, создастся запись пациента в базе данных после проверки введенной информации на корректность (валидации), созданная запись пациента будет выбрана для дальнейших действий. Также можно выбрать пациента, просмотрев посещения, запланированные на текущую дату.

После выбора пациента откроется основное окно приёма. Там содержатся поля для ввода информации о посещении, такой как жалобы, давление, диагноз и так далее.

На окне приёма будут присутствовать кнопки-переходы на следующие окна: «Выписка рецепта», «Назначить посещение». Эти окна выполняют функцию, соответствующую названию. Также на этом окне присутствует кнопка «Завершить приём», которая останавливает временной отсчёт и записывает всю информацию в базу данных.

Теперь подробнее о меню, присутствующем в окне начала приёма. В меню будут пункты, которые позволяют выводить различную информацию о работе терапевта: количество пациентов за день, общее время приёмов, количество выписанных рецептов. На всех окнах программы (кроме первого) будет возможность перейти на предыдущее, тем самым отменив запись всех данных, указанных на нём.

Администратор приложения будет иметь возможность просматривать учётные записи терапевтов, изменять их и создавать новые.

При входе в приложение с учётной записи пациента доступна одна возможность – запись на новый приём. Для записи необходимо с помощью полей поиска найти собственную запись в базе данных, выбрать желаемую дату и время приёма, а также терапевта. Информация о запланированном приёме заносится в базу данных. На рисунке 2.1 отображены пользователи разрабатываемого программного средства и действия, которые они могут выполнять.

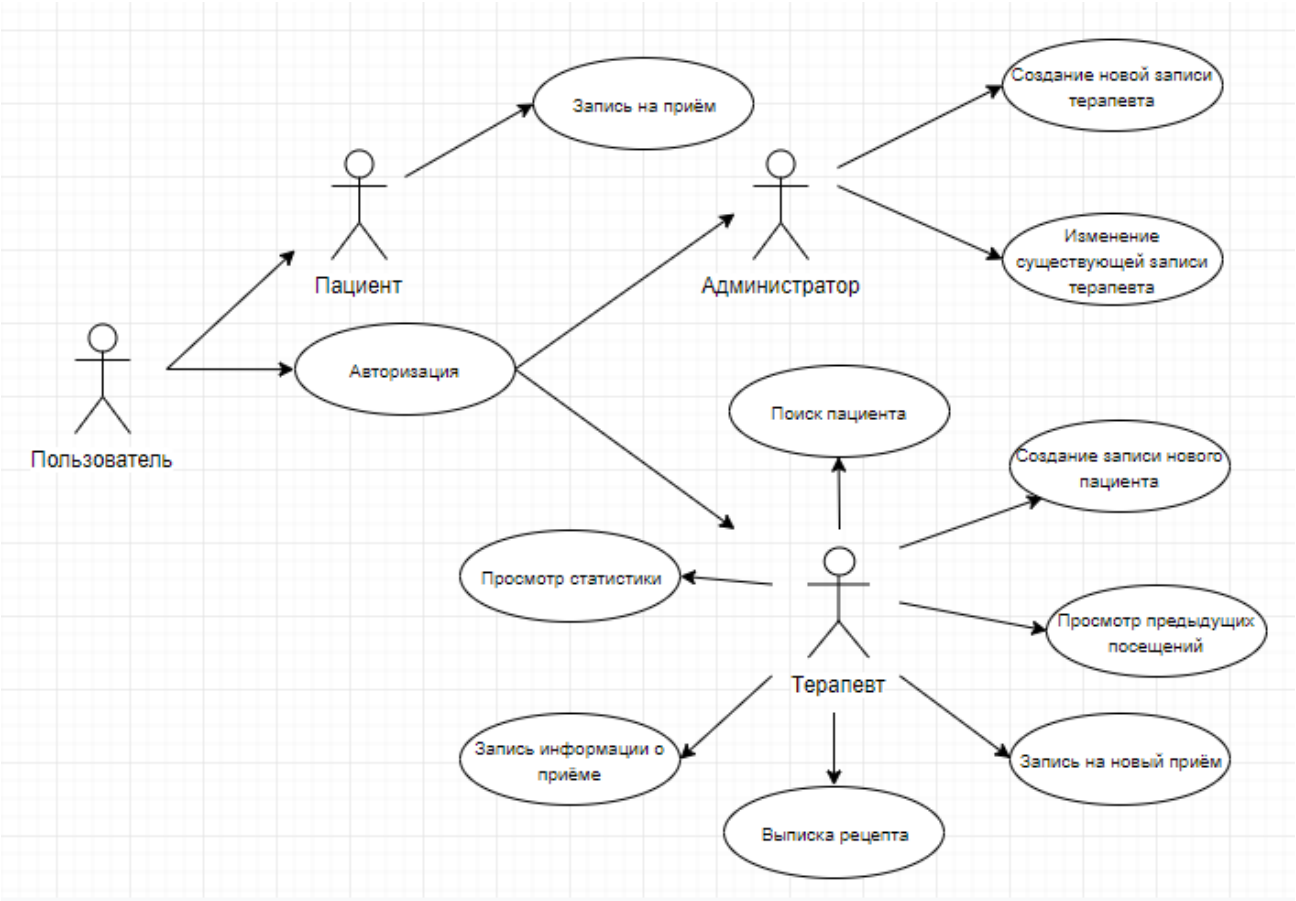


Рисунок 2.1 – Диаграмма использования

На основе данной UML-диаграммы и словесного описания функциональных требований строится вся бизнес логика программы.

3 Проектирование программного средства

В предыдущей главе были разработаны технические требования к программному средству. Для реализации разрабатываемого проекта была выбрана технология Windows Presentation Foundation, в качестве системы управления базами данных – Microsoft SQL Server.

В ходе проектирования были разработаны на языке Unified Modeling Language следующие диаграммы: диаграмма деятельности для участка программного средства, отвечающего за начало приёма терапевтом; диаграмма последовательностей для выбора пациента, диаграмма классов и предоставленная в предыдущей главе диаграмма вариантов использования. Разберём их подробнее.

Диаграмма классов (предоставлена в приложении А) отображает классы-сущности, используемые в программе. Основными классами являются классы PATIENT и VISIT, которые служат для хранения информации о соответственно пациентах и посещениях. Класс ADDRESS связан с классом PATIENT отношением ассоциации. Если уточнять, то PATIENT агрегирует ADDRESS. Это отношение было получено при помощи внешнего ключа в базе данных и соответственно полем ADDRESS_ID, имеющее целочисленный тип данных, указывающее на идентификатор адреса в таблице адресов, соответствующий нужной записи. Благодаря этой связи имея объект класса PATENT можно установить объект ADDRESS, в котором будет храниться информация о месте проживания данного пациента. Кроме этого, класс PATIENT содержит поля, необходимые для хранения информации о человеке, такой как фамилия, имя, отчество, пол, дата рождения, контактный телефон. Фамилия, имя и отчество ограничиваются длиной в 30 символов. Обязательными полями являются только фамилия, имя и ссылка на адрес проживания.

Класс ADDRESS в свою очередь хранит информацию об улице и доме проживания. Они имеют строковый тип данных. Улица имеет ограничение на длину в 64 символа, дом – в 5 символов. Дом имеет строковый тип данных, так как нередко существуют такие номера домов, как «14 А» или «14 В», если дом был построен уже после нумерации остальных домов улицы, а сделать адрес, по которому будет легко найти дом, необходимо. Также есть возможность добавить корпус или квартиру проживания, но они не являются обязательными. Поле ввода номера квартиры допускает ввод только цифр.

Класс RECIPE отвечают за информацию о рецепте. Он содержит такие поля как название медикамента, количество, идентификатор пациента и дата окончания действия рецепта. Обязательными являются поля медикамент и количество. Идентификатор пациента заполняется автоматически, потому что терапевт выбирает пациента и дальнейшие действия производятся с ним. Благодаря этому идентификатора образуется связь-ассоциация с классом PATIENT.

Класс VISIT агрегирует класс PATIENT. Это происходит благодаря наличию поля-идентификатора пациента. Также он содержит такие поля, как дата и время начала и конца посещения, жалобы, давление, текущий рост и вес пациента, диагноз и дополнительную информацию. Дата и время начала

посещения являются обязательным полем. Также существуют обязательные поля, показывающие, запланированный ли визит и состоялся ли он.

Кроме идентификатора пациента класс VISIT содержит ещё идентификатор пользователя. Это позволяет получать информацию о терапевте, который ответственен за это посещение. Эта информация хранится в объекте класса USERS. Данный класс содержит: логин, хэш пароля, ФИО терапевта и смену. Смена имеет только два допустимых значения (1 или 2). В зависимости от этого значения будут высчитываться часы приёма пациентов. С помощью логина и хэша пароля терапевт будет заходить в свою учётную запись. Все поля являются обязательными. Фамилия, имя и отчество терапевта ограничиваются длиной в 20 символов, логин – 15.

Каждый класс на диаграмме содержит идентификатор. Он целочисленный и заполняется автоматически при добавлении записи в базу данных при помощи оператора «identity». Все поля классов являются публичными.

Данная организация классов и базы данных позволяет получать всю информацию о визите, пациенте, его адресе проживания, выписанных ему рецептах, терапевте, имея объект класса VISIT, так как все классы связаны между собой отношениями-ассоциациями.

Каждое посещение содержит в себе ссылку на пользователя (терапевта) и пациента. Ссылка на пользователя получается благодаря авторизации. Пройдя эту процедуру, найденный в базе данных пользователь сохраняется в переменную и передается между последующими окнами в качестве параметра. Ссылка на пациента получается посредством выбора терапевтом либо запланированного визита (который уже содержит ссылку), либо непосредственно самого пациента, а далее опять же посредством параметров передаётся в последующие окна. То же самое происходит и со временем начала визита. Системное время записывается в переменную сразу после нажатия терапевтом кнопки «Начать визит», и передаётся в последующие окна. Когда будет нажата кнопка «Завершить визит», системное время запишется в переменную, отвечающую за время завершения посещения, и вся информация о посещении будет записана в базу данных.

3.1 Реализация посещений

Теперь подробно рассмотрим логику программы. На диаграмме деятельности, представленной в приложении Б, отображено начало приёма терапевтом, а конкретно процесс выбора пациента, с которым он ведёт работу в данный момент.

После начала приёма у терапевта есть несколько вариантов дальнейшей деятельности: либо просмотреть запланированные посещения на сегодня, либо выбрать пациента. Если он выбирает просмотреть запланированные посещения, то среди запланированных посещений он находит необходимое и выбирает его, процесс поиска текущего пациента завершен. Если терапевт решает выбрать пациента, то перед ним стоит вопрос: пациент пришёл впервые? Если нет, то терапевт приступает к поиску из списка. Если впервые, то врач создаёт новую запись пациента, а затем эта запись добавляется к этому списку. Если в списке найти пациента сложно, существуют поля ввода, где можно ввести имя, фамилию или дату рождения пациента, что сократит список пациентов и упростит поиск. Можно осуществлять поиск как по одному признаку, так и по всем сразу. После того, как пациент обнаружен в списке, терапевт выбирает эту запись. Теперь терапевт имеет возможность работать с текущей записью. Можно изменить информацию о пациенте или просмотреть предыдущие посещения, чтобы иметь доступ к его истории болезней. Также можно перейти к следующей фазе приёма – записи информации о самом посещении. Процесс поиска текущего пациента завершён.

Если в процессе поиска текущего пациента терапевт выбирает его из запланированных посещений, то данное посещение будет сохраняться в базе данных как запланированное. Если терапевт выбирает пациента из списка, то посещение автоматически становится незапланированным.

На окне записи на приём можно выбрать предпочтительного терапевта, дату и свободное время приёма. В выходные дни поликлиники не работают, поэтому при нажатии на дату выходного дня свободного времени не будет предлагаться. Вычисление свободного времени происходит следующим образом. Вычисляются часы приёма терапевта (если терапевт работает в первую смену, то часы приёма с 8 до 16, иначе с 14 до 22). Затем получаются все возможные времена приёма (интервал между записями – 15 минут), из базы данных выгружаются все используемые времена на выбранную дату. Происходит проверка всех возможных времён приёма, и если текущего времени нет среди запланированных записей, то оно выводятся.

Поиск пациента допускается осуществлять по трём критериям: фамилия, имя и дата рождения. Поиск по фамилии и имени организован на основе регулярных выражений. Если терапевт введет часть слова, то запись будет найдена.

Выбрав запланированное посещение или пациента, терапевт имеет возможность выписать рецепт больному на лекарство. Введенные данные проходят валидацию, информация о рецепте заносится в базу данных, а затем

предлагается вывести его на печать. Валидация заключается в проверке полей, отвечающих за медикамент и его количество на наличие информации. Если поля пустые ввода, то валидация не пройдена и выводится сообщение об этом. Также можно назначить повторное посещение для текущего пациента, чтобы иметь возможность контролировать протекание его болезни. На окне посещения терапевту будут предложены следующие поля для заполнения: диагноз, давление, жалобы, рост и вес, дополнительная информация. Обязательными они не являются, поэтому даже со всеми пустыми полями терапевт может завершить приём, если пациент, например, пришёл не по болезни, а с целью получить справку о текущем состоянии здоровья или новый рецепт.

После окончания посещения поле, отвечающее за завершенность, устанавливается в значение истина, и запись о посещении сохраняется в базе данных.

Закончив посещение, терапевту открывается первое окно после авторизации. На данном окне терапевт имеет следующие возможности: просмотреть свою статистику за текущую дату, просмотреть информацию о программе, выйти из своей учётной записи и начать следующее посещение. В статистике за сегодняшний день терапевт может видеть общее количество завершенных посещений за текущую дату, количество запланированных завершенных и незавершенных визитов, и суммарное время, потраченное на посещения. В информации о программе будет указано название программного средства, дата создания и имя разработчика приложения.

На диаграмме последовательности, предоставленной в приложении В, отображен процесс выбора пациента. Первоначально перед терапевтом стоит задача ввести данные для поиска. Из полей ввода данных информация идёт на дальнейшую обработку. В процессе обработки происходит валидация: проверка на наличие информации в поле и на корректность введённой даты, если она введена. После прохождения валидации происходит поиск в базе данных записей, подходящих условиям поиска. Результат выводится в соответствующий элемент управления и осуществляется выбор искомого пациента.

3.2 Работа с базой данных

В разрабатываемом программном средстве существует возможность изменения записей в базе данных.

Допускается редактирование существующих записей пациентов и создание новых, если пользователь вошел под учётной записью терапевта. При выборе пациента и нажатия на кнопку «Изменить информацию» откроется новое окно с уже заполненными текущей информацией о пациенте полями. При нажатии кнопки «Новый пациент» терапевту открывается такое же окно, как и при редактировании, но все поля будут незаполненными. Обязательная информация для заполнения: фамилия, имя, дата рождения, улица и дом проживания. Дополнительно можно указать контактный телефон, квартиру и корпус дома, пол. По умолчанию пол выставляется в значение «м».

Запись на приём можно организовать как с учётной записи терапевта, так и пациенту. Если пациент хочет записаться, то необходимо пройти процесс авторизации. Пользователю открывается окно, в котором существует несколько полей ввода для поиска собственной записи в базе данных. Это окно отличается от окна поиска для терапевта, отсутствуют такие функции, как просмотр предыдущих посещений, не допускается изменение и создание новой записи. Также пациент не имеет доступа к базе данных: пока пациент не заполнит хотя бы одно поле ввода, элемент управления, в котором отображается результат поиска, будет пуст. На подобном окне же терапевт сразу видит все записи пациентов из базы данных. Реализация работы с базой данных терапевта подробно описана в пункте 3.1.

В приложении существует единственная учётная запись терапевта. Он обладает правом создавать и редактировать записи терапевтов. При создании учётной записи терапевта открывается окно с пустыми полями для заполнения. Среди них ФИО терапевта, смену, логин и пароль. По умолчанию смена считается первой. Также есть недоступное для редактирования поле «хэш пароля», которое автоматически вычисляется при изменении пароля. В базу данных заносятся ФИО, смена, логин и хэш пароля. Сам пароль в базу не вносится в целях безопасности. Все значения являются обязательными.

Во всех окнах, где требуется вывод записей из базы данных, информация заносится в элемент управления DataGrid, столбцы получают данные при помощи реализации привязки.

4 Реализация программного средства

4.1 Структура проекта

На рисунке 4.1.1 отображена структура разрабатываемого программного средства.

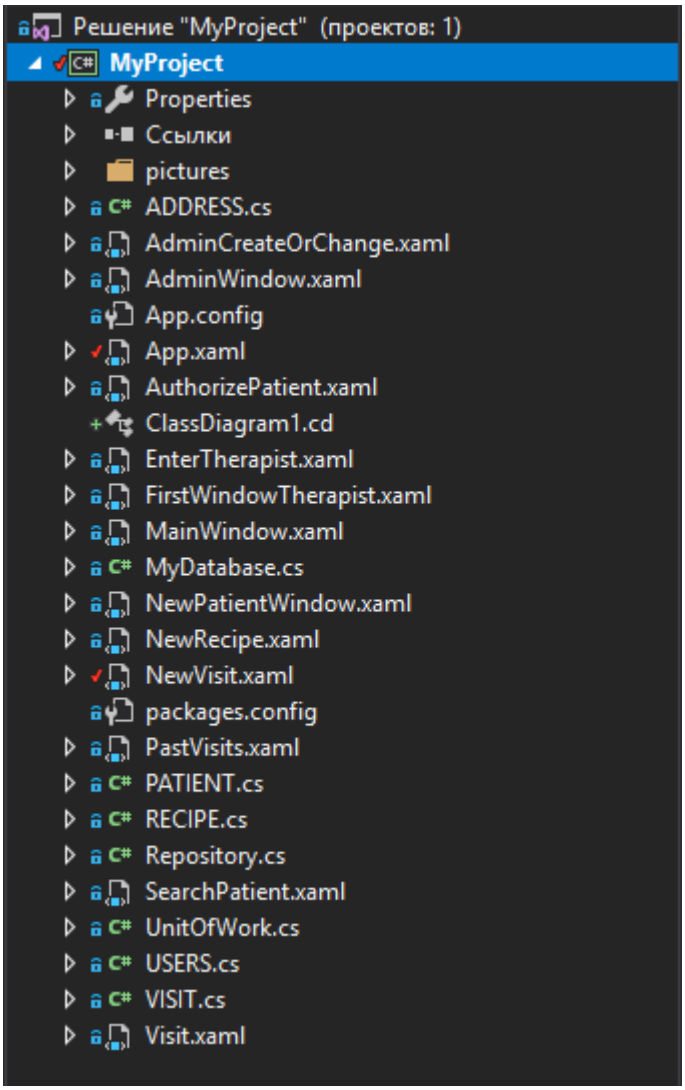


Рисунок 4.1.1 – Структура проекта

В проекте присутствуют классы-сущности, окна (WPF), диаграмма классов, некоторые ресурсы и файл конфигурации, реализация паттернов Repository и UnitOfWork.

К классам-сущностям относятся:

– ADDRESS, объект этого класса хранит информацию об адресе пациента;

– PATIENT, используется для определения данных о пациенте;

– VISIT, содержит поля, отвечающие за данные посещения;

– RECIPE, представляет собой класс для хранения информации о выписанных рецептах

– USERS, содержит информацию о пользователе.

Данные классы реализуют технологию EntityFramework.

За реализацию паттернов Repository и UnitOfWork отвечают одноименные файлы. Для каждой сущности был создан класс-репозиторий, который реализует интерфейс IRepository<T>, где обобщенный тип принимает значение соответствующего класса-сущности. Методы интерфейса IRepository<T> и список классов-репозиториев предоставлен на рисунке 4.1.2.

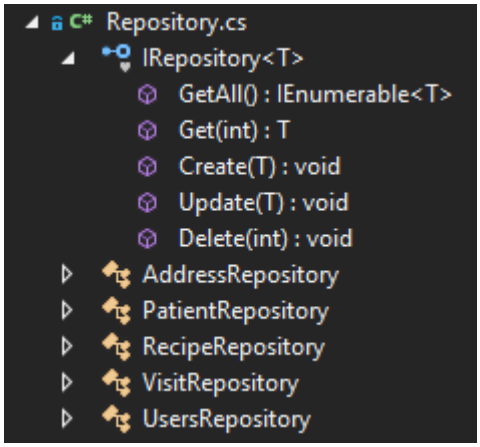


Рисунок 4.1.2 – Содержание файла Repository.cs

В файле App.config хранятся параметры подключения к базе данных.

Папка pictures хранит в себе изображения, являющиеся иконкой приложения или фоном окон.

ClassDiagram1.cd – диаграмма классов, представленная на рисунке 4.1.3.

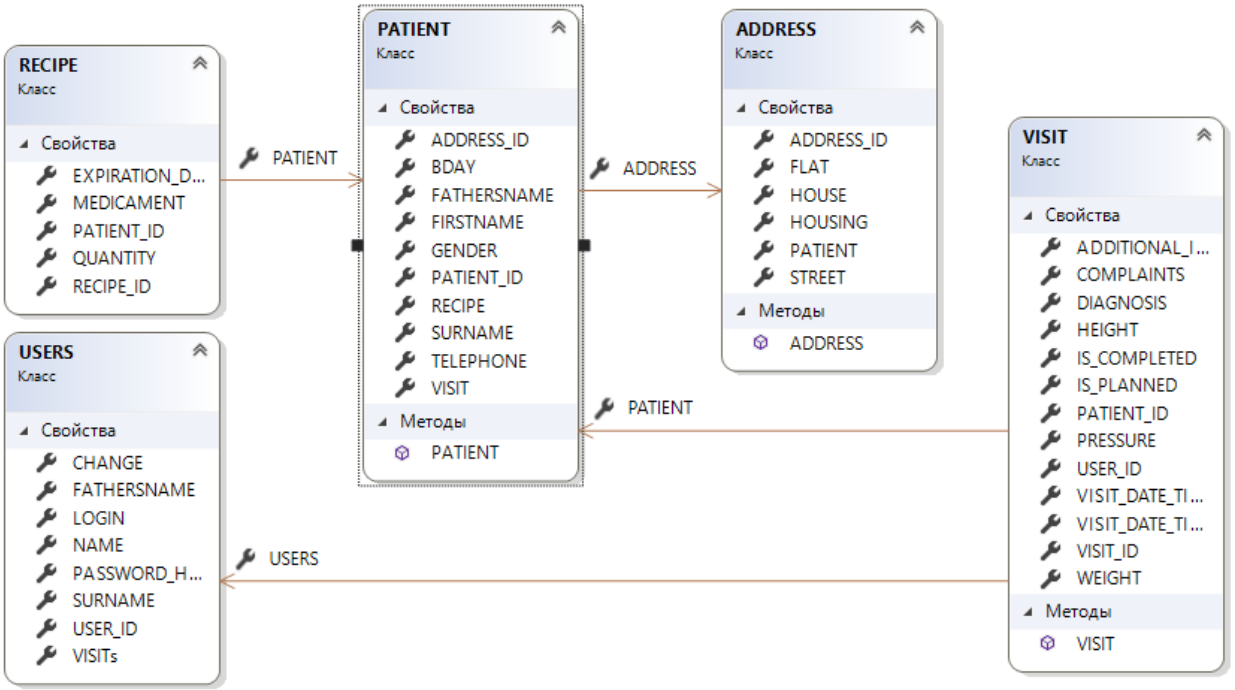


Рисунок 4.1.3 – Диаграмма классов

Данная диаграмма классов соответствует диаграмме, разработанной в пункте 3.

Файл MyDatabase.cs представляет собой модель.

Окна составляют основу дизайна и обработки данных. В разрабатываемом программном средстве присутствуют следующие окна:

– AdminCreateOrChange, окно администратора, вызываемое при создании или изменении записи терапевта;

– AdminWindow, окно администратора, где показываются записи терапевтов, есть кнопки для изменения или создания новой записи;

– AuthorizePatient, предоставляет пациенту возможность найти собственную запись в базе данных для дальнейшей работы с ней;

– EnterTherapist, окно авторизации, в зависимости от логина и пароля заходит в учётную запись терапевта или администратора;

– FirstWindowTherapist, главное окно терапевта, на нем присутствуют кнопка начала приёма, меню;

– MainWindow , стартовое окно приложения;

– NewPatientWindow, окно создания или изменения записи пациента;

– NewRecipe, окно, для создания нового рецепта и дальнейшей отправки его на печать;

– NewVisit, окно для записи на посещение, к нему имеют доступ как пациенты, так и терапевты;

– PastVisits, окно для отображения предыдущих посещений конкретного пациента либо для предоставления информации о запланированных на текущий день визитах;

– SearchPatient, окно выбора пациента для проведения с ним приёма;

– Visit, окно посещения, содержит поля для ввода информации о приёме.

В файлах .xaml окон содержится визуальное оформление проекта, в файлах расширения .cs – обработка данных

4.2 Реализация добавления записей

В разрабатываемом программном средстве есть возможность добавлять записи в базу данных. Терапевт имеет возможность добавлять в базу новых пациентов, новые посещения и создавать рецепты. С учётной записи администратора есть возможность создавать новые записи терапевтов. Пациент имеет возможность лишь регистрироваться на новые посещения, они добавляются в базу данных, но имеют статус незавершенных.

В общем виде добавление записей в базу данных осуществляется следующим образом:

– пользователь заполняет поля для ввода информации;

– пользователь нажимает на кнопку подтверждения;

– проверяется заполнение обязательных полей. Если поля не заполнены, то выводится соответствующее сообщение, информация не заносится в базу данных;

– проводится валидация данных. В основном на корректность проверяются поля для ввода даты. Если строку даты невозможно привести к типу DateTime, выводится сообщение об ошибке, информация в базу не заносится;

– если все данные валидны, создаётся объект класса-сущности, информация из полей ввода заносится в соответствующие свойства объекта;

– объект с помощью экземпляра класса UnitOfWork методом Create() помещается в базу данных;

– объект UnitOfWork производит сохранение изменений методом Save().

Этот алгоритм применяется при каждом добавлении записи в базу данных.

4.3 Реализация поиска

Поиск в разрабатываемом программном средстве допускается только по пациентам.

Терапевт, начав приём, должен найти пациента из общего списка, выделить его и перейти на следующее окно. Также, пациент, при записи на приём, должен найти собственную запись в базе данных, если она существует.

Приложение реализует поиск по фамилии и имени на основе регулярных выражений. Поля ввода данных изначально проверяются на наличие информации. Если строка не пустая, то данные из текстового поля заносятся в шаблон, в котором до данного текста и после может быть любая последовательность символов. Сначала выполняется поиск по фамилии, затем отобранный список проходит отсеивание по имени, а после этого по дате рождения. Если поле ввода пустое, то поиск по соответствующему свойству пропускается. В зависимости от учётной записи, при всех пустых полях будут выводиться различные результаты. Если был выполнен вход в запись терапевта, то при пустых полях поиска будет выводиться полный список пациентов. Если окно открыто без выполнения авторизации (то есть вход как пациент), то при всех пустых полях поиска не выведется ни одна запись.

Сам поиск реализуется следующим образом. В переменную-список загружаются все объекты PATIENT из базы данных. Далее, исходя из введенной информации, происходит отсев загруженных данных. Если поиск осуществляется на окне терапевта, то непрошедшие фильтр записи удаляются. Если это поиск с окна для пациента, то подходящие записи добавляются в элемент DataGrid. Информация в данном элементе отображается при помощи привязок.

4.4 Реализация создания и печати рецепта

Для создания нового рецепта существует отдельное окно со следующими полями для заполнения: медикамент, количество, срок годности рецепта. Они все являются обязательными для начала реализации печати. Срок годности рецепта также проходит валидацию. Если дату не удаётся преобразовать к типу DateTime или если указанная дата уже прошла, выводится соответствующее сообщение.

Создание и печать реализуется следующим образом:

– проводится валидация значений в полях ввода;

– создаётся новый объект класса RECIPE;

– в свойства объекта помещаются введенные значения;

– вызывается метод Create() свойства Recipes объекта UnitOfWork с созданным ранее объектом класса RECIPE, сохраняется изменение базы данных вызовом метода Save объекта UnitOfWork;

– создаётся объект класса PrintDialog;

– если метод printDialog.ShowDialog() прошёл успешно, то производятся следующие действия;

– в объекты класса Run помещается текст и информация из полей ввода;

– настраивается отступ, поля, вертикальная ориентация текста;

– каждый из объектов помещаencz в отдельный TextBox;

– создаём объект класса stack;

– настраивается ориентация стека (вертикальная);

– последовательно в стек помещаются созданные текстбоксы;

– вызывается метод PrintVisual объекта PrintDialog со следующими параметрами: созданный стек и строка-название открывающегося окна, в котором можно будет выбрать принтер или конвертировать рецепт в формат PDF.

При выборе принтера или желаемого действия и начатии кнопки «ОК» действие будет произведено, окно создания нового рецепта закроется.

5 Тестирование, проверка работоспособности и анализ полученных результатов

5.1 Соответствие техническому заданию

Разработанное программное средство полностью соответствует составленному для него техническому заданию. Был реализован весь описанный функционал, также добавлены некоторые дополнительные функции.

Программа функционирует и компилируется без ошибок.

В качестве проверки корректного функционирования программы реализовала приём пациента, пришедшего на приём впервые. На рисунке 5.1.1 представлено окно добавления новой записи.

Рисунок 5.1.1 – Создание новой записи пациента

После нажатия на кнопку «Создать» пациент был добавлен в базу данных (рисунок 5.1.2).

	PATIENT_ID	SURNAME	FIRSTNAME	FATHERSNAME	GENDER	BDAY	TELEPHONE	ADDRESS_ID
5	5	Слюсарь	Владислав	Игоревич	м	1997-09-26	+375982911616	2
6	6	Пантилимонов	Богдан	Витальевич	м	1997-09-01	+375677962881	4
7	7	Кухтина	Диана	Валерьевна	ж	1998-03-30	+375681887710	7
8	8	Хмиль	Дмитрий	Витальевич	м	1994-09-06	+375201527318	8
9	9	Ковальчук	Максим	Владимирович	м	1994-06-01	+375662313753	9
10	10	Петренко	Александр	Григорьевич	м	1996-08-30	+375552726229	10
11	11	Шевченко	Пётр	Алексеевич	м	1995-02-26	+375392997210	11
12	12	Некрасова	Юлия	Викторовна	ж	1999-05-09	+375211923544	12
13	13	Квитко	Анастасия	Олеговна	ж	1998-05-12	+375432397733	13
14	14	Волочкова	Анастасия	Юрьевна	ж	1996-01-20	+375241429114	14
15	15	Глухаревский	Игорь	NULL	м	2001-10-10	+375292847122	15

Рисунок 5.1.2 – Добавленная запись в базе данных

Пациент также появился в общем списке всех пациентов. Выбираем его из списка для проведения визита. Заполняем поля информацией о посещении (рисунок 5.1.3).

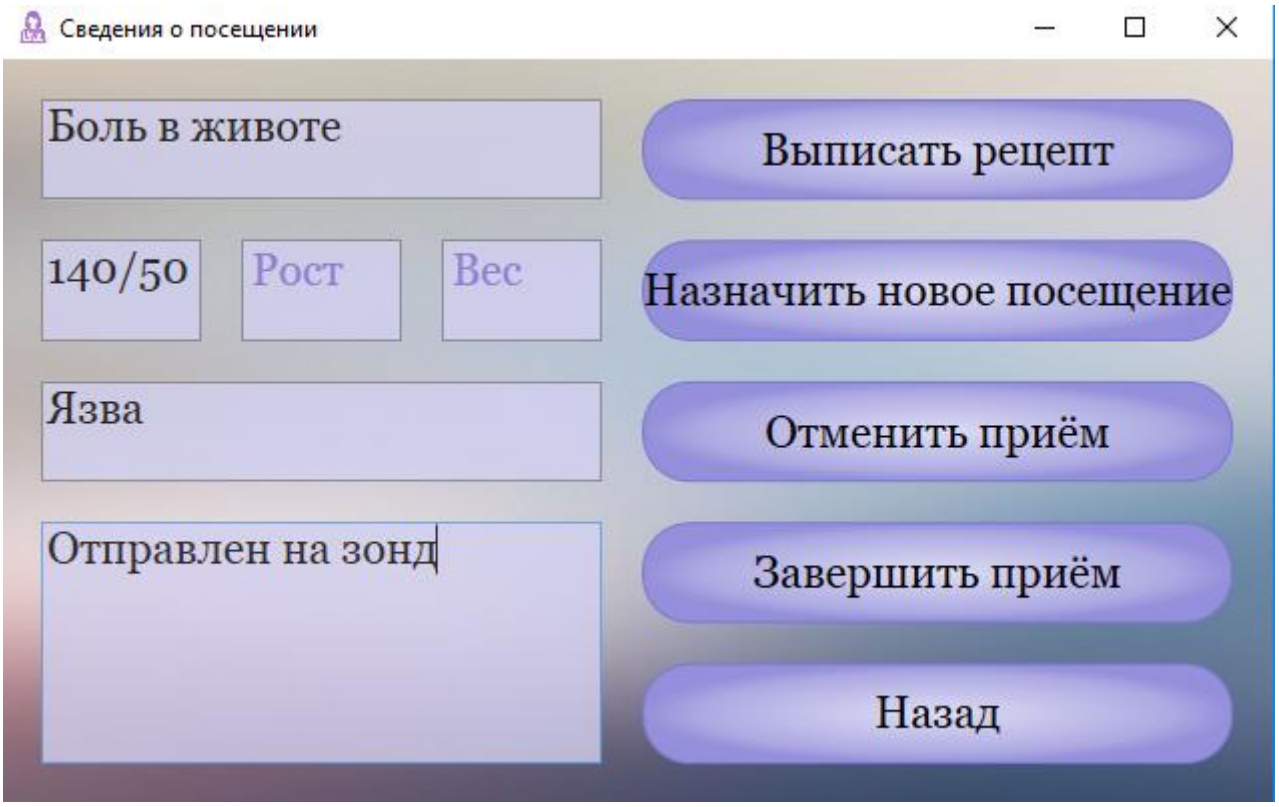


Рисунок 5.1.3 – Создание записи посещения

Создаём новый рецепт. Для этого требуется нажать на кнопку «Выписать рецепт» и заполнить обязательные поля (рисунок 5.1.4).

Рисунок 5.1.4 – Создание рецепта

Нажимается кнопка «Ок», пользователь работает с диалоговым окном печати, создаётся объект класса-сущности и помещается в базу данных, затем происходит возврат к предыдущему окну. Терапевт завершает приём и вся

информация заносится в базу данных. На рисунке 5.1.5 предоставлены результаты записи в базу данных.

	VISIT_ID	PATIENT_ID	VISIT_DATE_TIME1	COMPLAINTS
7	7	6	2019-05-20 13:00:00.000	Повышенная температура
8	8	7	2019-05-20 13:15:00.000	Повышенная утомляемость
9	9	8	2019-05-20 10:15:00.000	Повышенная температура, каше
10	10	9	2019-05-20 11:30:00.000	Повышенная утомляемость
11	11	10	2019-05-23 12:15:00.000	Повышенная утомляемость
12	12	11	2019-05-23 12:45:00.000	Повышенная температура
13	13	12	2019-05-23 12:30:00.000	Повышенная утомляемость
14	14	13	2019-05-23 11:00:00.000	Повышенная температура, каше
15	15	14	2019-05-23 09:15:00.000	Боль в спине
16	16	15	2019-05-21 15:20:06.267	Боль в животе

	RECIPE_ID	MEDICAMENT	QUANTITY	EXPIRATION_DATE	PATIENT_ID
1	1	Карвалол	20 таблеток	2019-09-23	2
2	2	Фармил	30 мг	2019-06-10	1
3	3	Арвалин	15 таблеток	2019-07-21	15

Рисунок 5.1.5 – Добавленные в базу данных записи

Таким образом, основную свою функцию приложение выполняет и соответствует техническому заданию.

5.2 Тестирование программного средства

В разработанном программном средстве предусматриваются многие исключительные ситуации.

Создавая новую запись, предназначенную для помещения в базу данных, при незаполненных обязательных полях на экран выводится сообщение, продемонстрированное на рисунке 5.2.1.

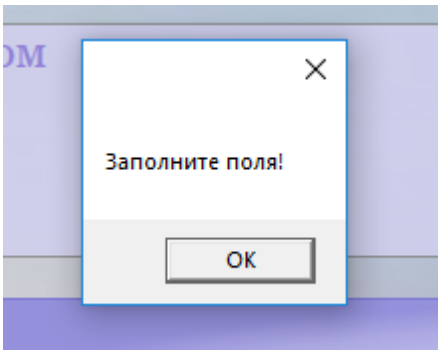


Рисунок 5.2.1 – Сообщение, возникающее при пустых обязательных полях

На окнах, в которых предполагается выбор одного пациента из нескольких, содержащихся в DataGrid, при попытке произведения действий, для которых предполагается наличие выделенной записи, а такой нет, выводится сообщение, отображенное на рисунке 5.2.2.

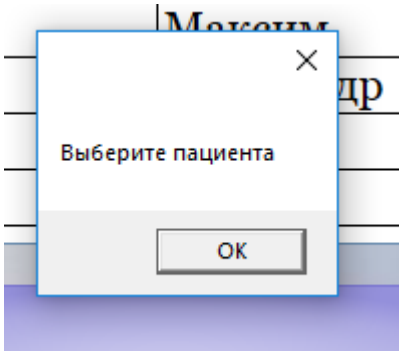


Рисунок 5.2.2 – Сообщение о необходимости выбора записи

В приложении присутствует аутентификация пользователя. В базе данных хранятся допустимые комбинации значений логина и пароля. Если при входе в систему указываются неверные данные, выводится сообщение, продемонстрированное на рисунке 5.2.3.

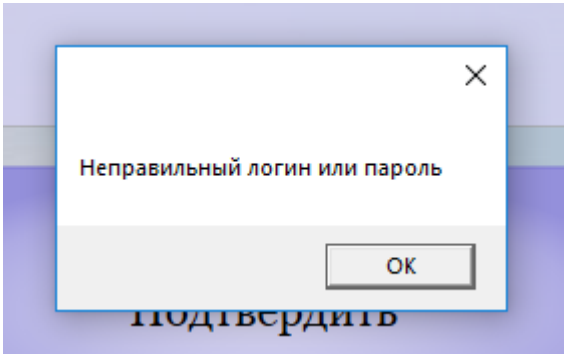


Рисунок 5.2.3 – Сообщение о неправильно введенных данных пользователя

При попытке записи на новое посещение, если пользователь не выбрал хотя бы одно обязательное значение, будет выведено сообщение, представленное на рисунке 5.2.4.

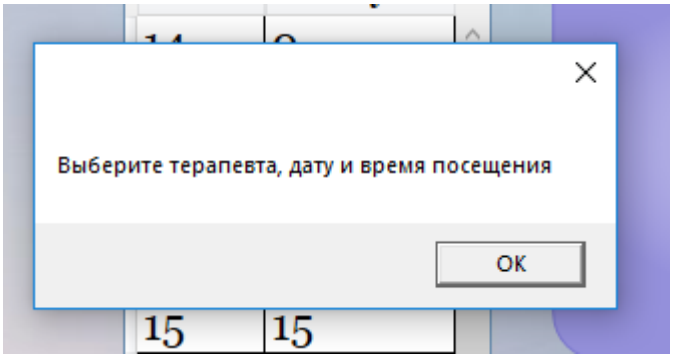


Рисунок 5.2.4 – Сообщение о недостаточности введенных данных

Поля ввода даты содержат валидацию. При некорректно введенной дате выводится сообщение, продемонстрированное на рисунке 5.2.5.

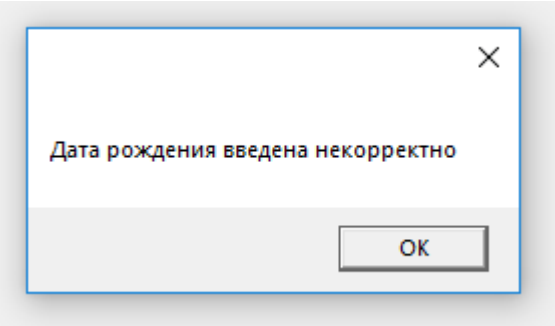


Рисунок 5.2.5 – Сообщение о некорректности введенных данных

Таким образом, программное средство прошло разработанный в данной главе ограниченный набор тестов. Результаты тестирования оказались успешны, ни в одной ситуации приложение не создало неконтролируемого исключения. Можно сделать вывод, что программное средство работает корректно и готово к использованию.

6 Руководство по установке и использованию

Для корректной работы приложения необходимо наличие следующих компонентов:

- компьютер со всеми необходимыми для его работы компонентами (процессором, видеокартой, оперативной памятью);
- установленный и работающий SQL Server 2012;
- операционная система Windows 10;
- Visual Studio 2019;
- Entity Framework 6.0.0;

Для работы с базой данных необходимо запустить файл createDB.sql. Данный файл содержит в себе скрипт создания необходимой для программного средства базы данных и небольшое количество данных, которыми заполняются таблицы.

Если выполнение всех инструкций sql файла прошло успешно и все необходимые ресурсы и программы имеются, можно начинать работу с приложением. Текст файла created.sql находится в приложении Г.

6.1 Руководство для администратора

В приложении существует лишь одна запись администратора. Для входа как администратор требуется на стартовом окне нажать кнопку «Войти», затем в появившемся окне указать в поле «Логин» значение «admin» и ввести верный пароль.

Главное окно терапевта представлено на рисунке 6.1.1.

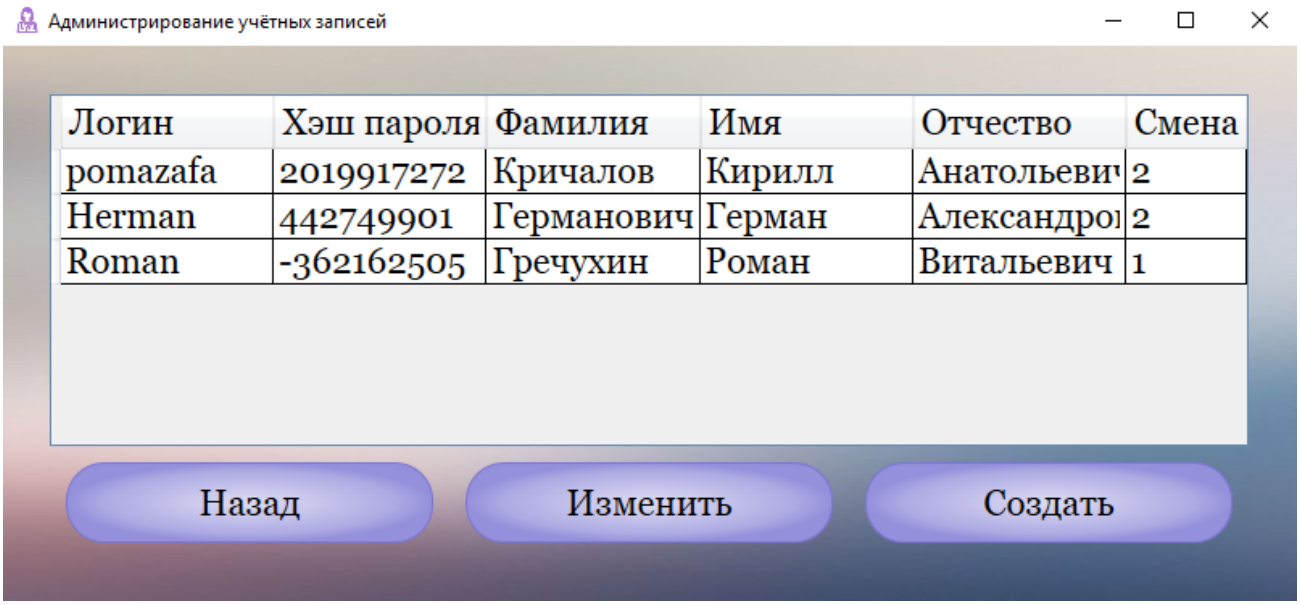


Рисунок 6.1.1 – Главное окно терапевта

В данном окне отображаются записи всех терапевтов, имеющиеся в базе данных. Терапевт имеет возможность создавать новые записи или редактировать уже существующие. Для создания новой записи необходимо нажать на кнопку «Создать». Откроется окно, в котором требуется заполнить все поля. После заполнения всех полей необходимо нажать «Ок» и созданная запись отобразится на окне, представленном на рисунке 6.1.1. Чтобы изменить существующую запись, требуется выделить интересующую запись и нажать на кнопку «Изменить». В открывшемся окне изменить информацию на новую и также нажать «Ок». Если администратору необходимо вернуться на предыдущее окно, необходимо нажать кнопку «Назад».

6.2 Руководство для пациента

Чтобы использовать приложение как пациент, необходимо на первом окне программного средства нажать «Войти как пациент».

В открывшемся окне, представленном на рисунке 6.2.1, необходимо с помощью поиска по фамилии, имени и дате рождения найти собственную запись в базе данных.

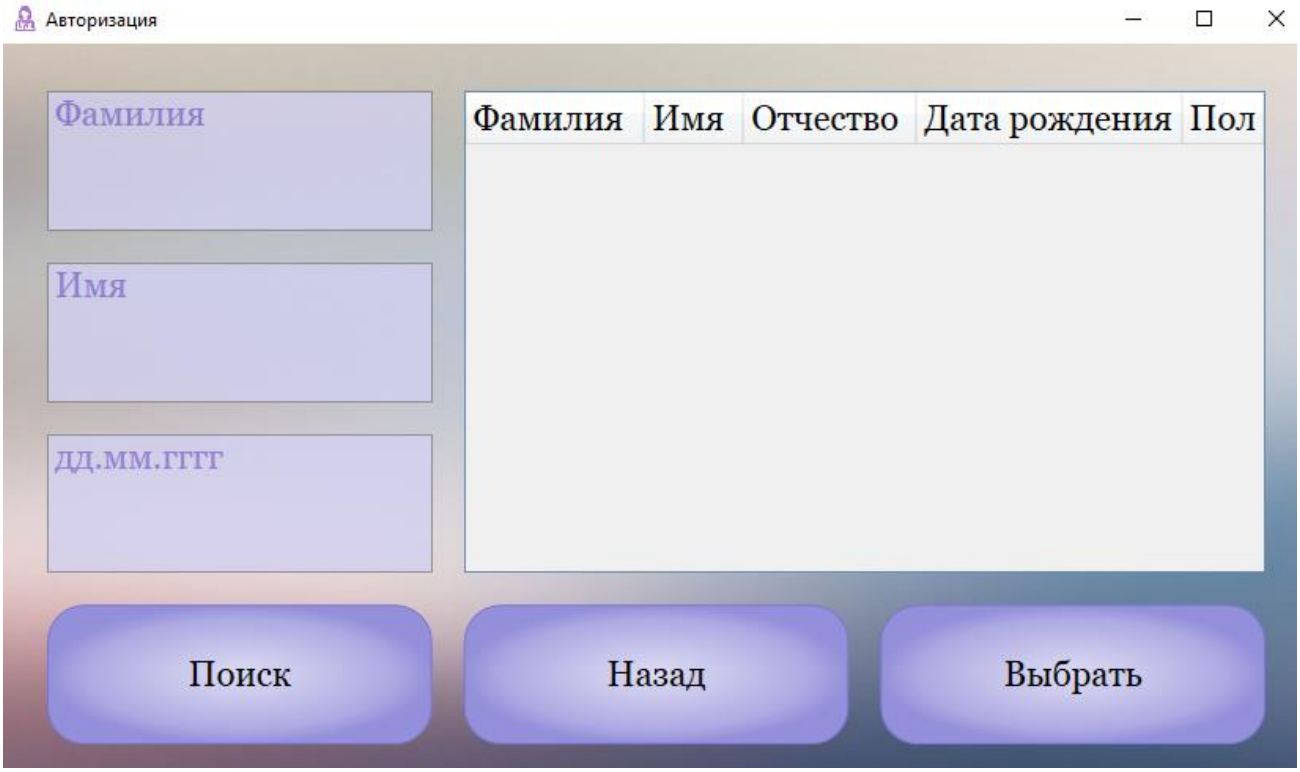


Рисунок 6.2.1 – Окно поиска записи пациента

Для поиска необходимо заполнять поля Фамилия, Имя и Дата рождения верными данными о себе. После этого требуется нажать «Поиск». Если запись найдена, требуется выделить её и нажать «Выбрать» для перехода на следующее окно. Если по какой-либо причине пользователь передумал записываться на приём, он может нажать на кнопку «Назад» и вернуться к первому окну.

После выбора записи, пользователю открывается окно записи на приём (рисунок 6.2.2). В данном окне пациенту необходимо выбрать терапевта, дату и время приёма.

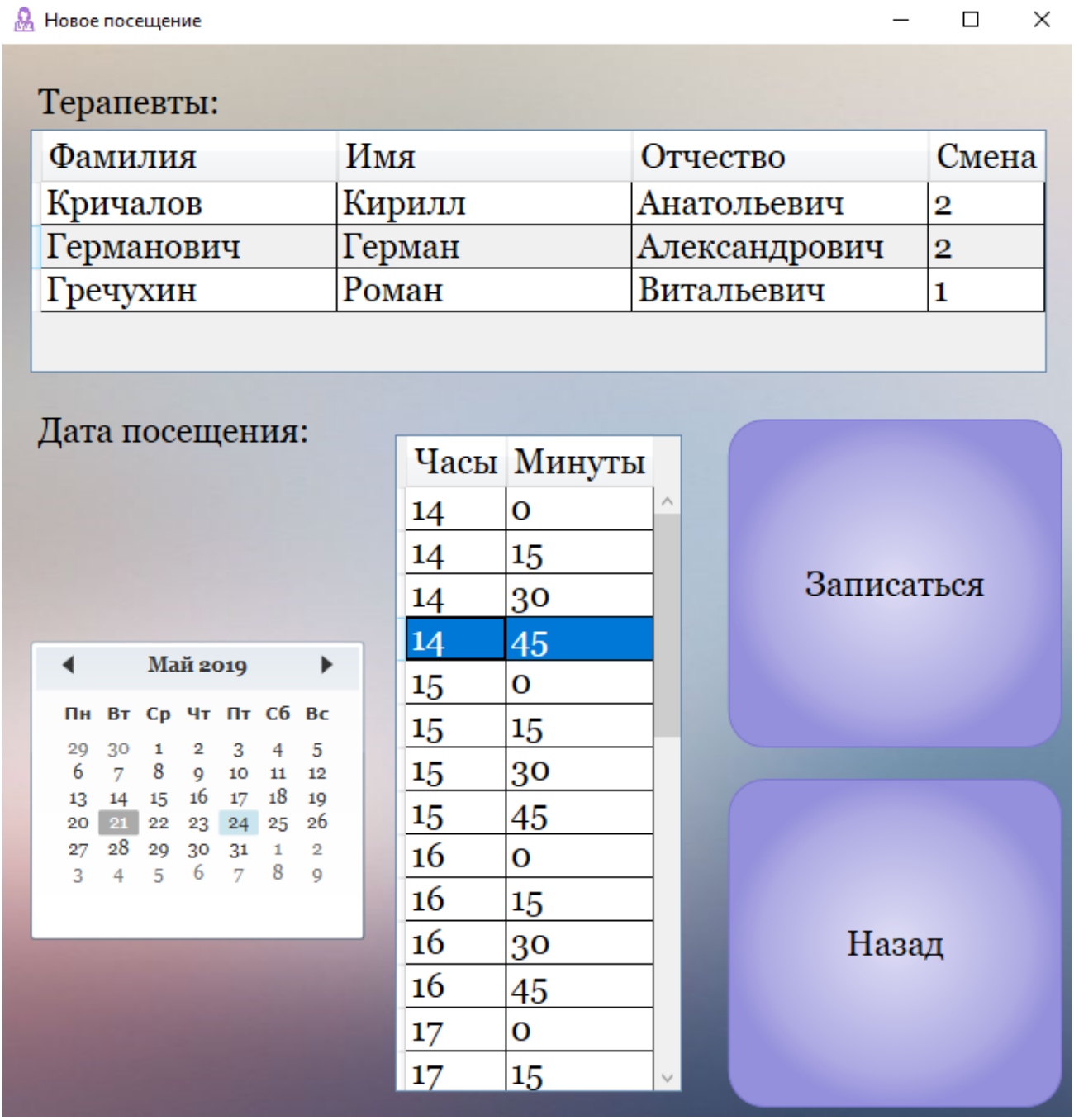


Рисунок 6.2.2 – Запись на приём

При выборе прошедшей даты либо выходных дней допустимое время не будет отображаться. При выборе корректной даты и терапевта, высчитываются часы работы терапевта и появляется доступное время. Выбрав все пункты, пациент должен нажать «Записаться», чтобы его запись сохранилась в базе данных. Затем пациенту открывается главное окно и выводится сообщение об успешной записи на приём. Если же пациент не хочет записываться на приём, он должен нажать на кнопку «Назад». Это весь функционал, доступный пациенту.

6.3 Руководство для терапевта

Для входа в учётную запись терапевта на первом окне приложения необходимо нажать кнопку «Войти». Далее в полях ввода ввести верные логин и пароль.

После ввода верных данных, откроется главное окно терапевта, представленное на рисунке 6.3.1.

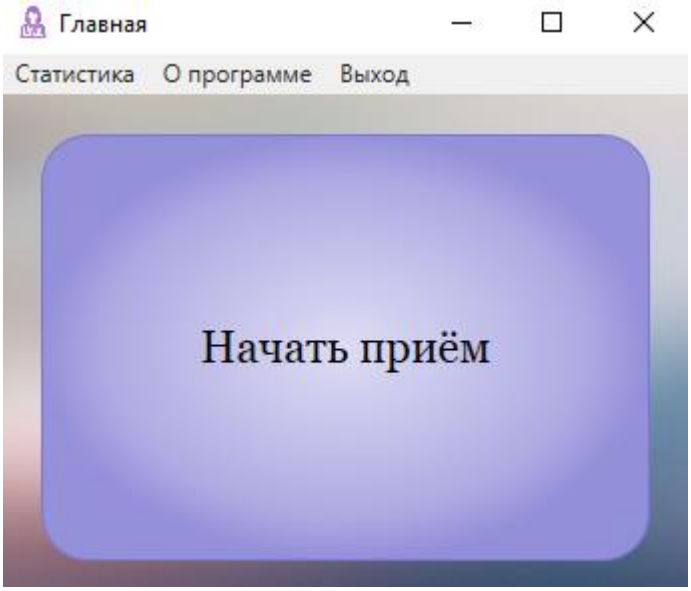


Рисунок 6.3.1 – Главное окно терапевта

На данном окне присутствует меню, в котором есть следующие пункты: просмотр статистики (при нажатии отображается статистика терапевта за текущий день), «О программе» и выход. При нажатии на «Выход» открывается первое окно приложение.

Кроме меню присутствует единственная кнопка «Начать приём». При нажатии на неё откроется окно поиска пациента, представленное на рисунке 6.3.2.

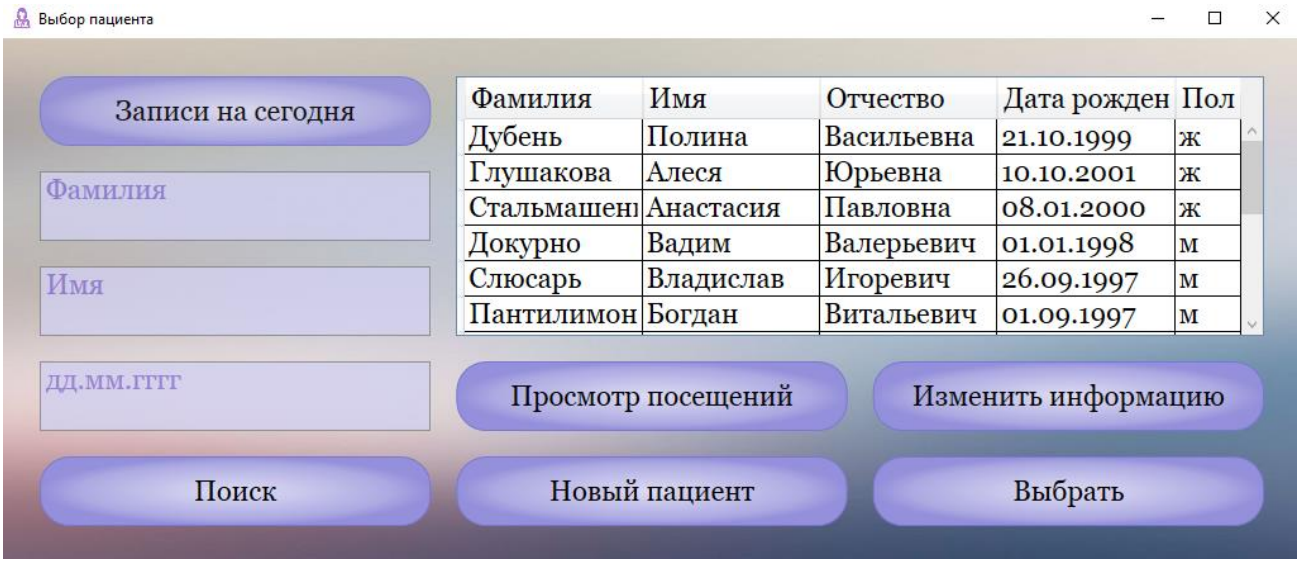


Рисунок 6.3.2 – Окно выбора пациента

Далее у терапевта присутствует несколько вариантов действий. Они представлены на схеме, которая размещена в приложении Б. Чтобы просмотреть посещения на сегодня, необходимо нажать на кнопку «Записи на сегодня. Откроется окно, в котором будут выведены незавершенные запланированные на сегодня к текущему терапевту посещения. Окно имеет вид, представленный на рисунке 6.3.3.

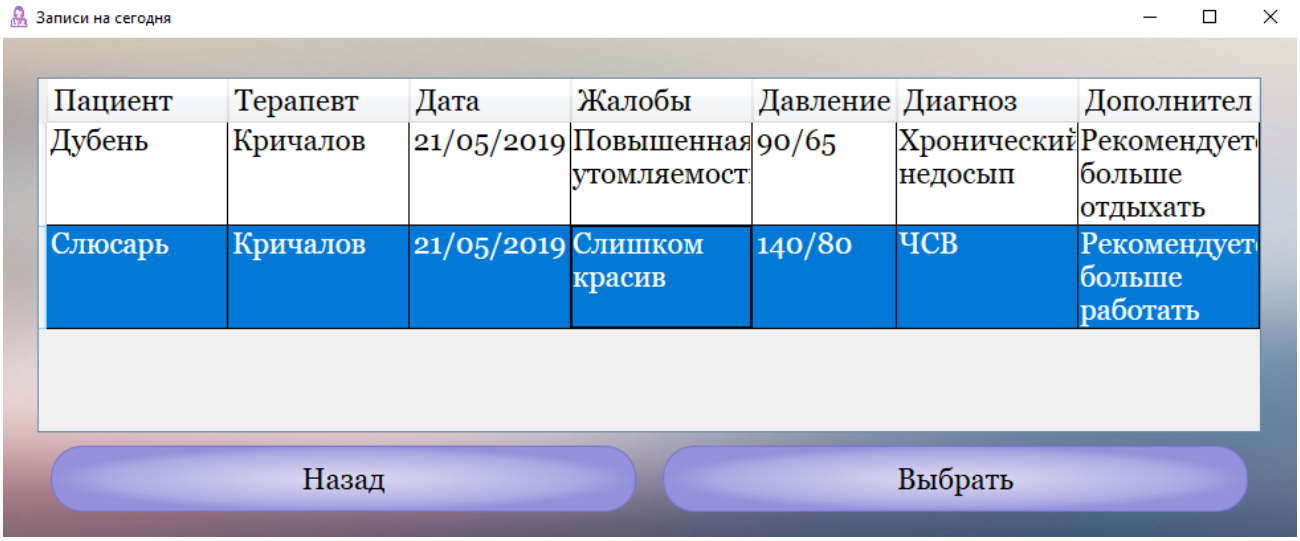


Рисунок 6.3.3 – Записи на текущий день

Чтобы выбрать посещение для работы с ним, требуется выделить его в списке и нажать кнопку «Выбрать». Если посещения с нужным пациентом не нашлось, можно нажать на кнопку «Назад» и приложение вернется на предыдущее окно.

Если пациент пришёл к терапевту впервые, можно создать новую запись пациента. Для этого требуется нажать кнопку «Новый пациент». Откроется окно, представленное на рисунке 6.3.4.

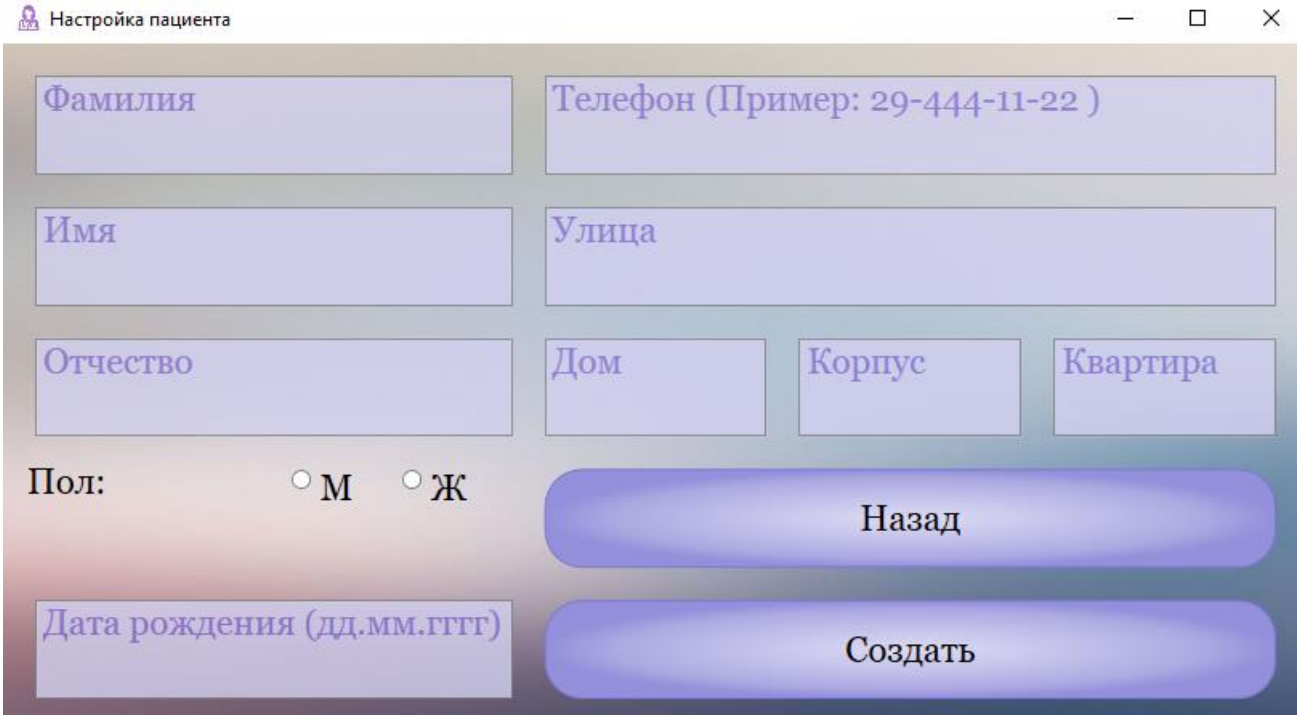


Рисунок 6.3.4 – Создание нового пациента

Терапевту требуется заполнить обязательные поля и нажать кнопку «Создать». Если данные прошли валидацию, запись будет занесена в базу данных и будет отображаться на окне 6.3.3. При нажатии кнопки «Назад» данное окно закрывается и открывается предыдущее. Окно на рисунке 6.3.4 открывается также при нажатии на кнопку «Изменить». Поля ввода будут заполнены информацией о выделенном пациенте, если таковой имеется. Также присутствует возможность выделить пациента и просмотреть его историю посещений. Для этого необходимо нажать на кнопку «Предыдущие посещения». Если терапевт выбрал пациента или посещение, он может нажать на кнопку «Выбрать», перейдя к окну, представленном на рисунке 6.3.5.

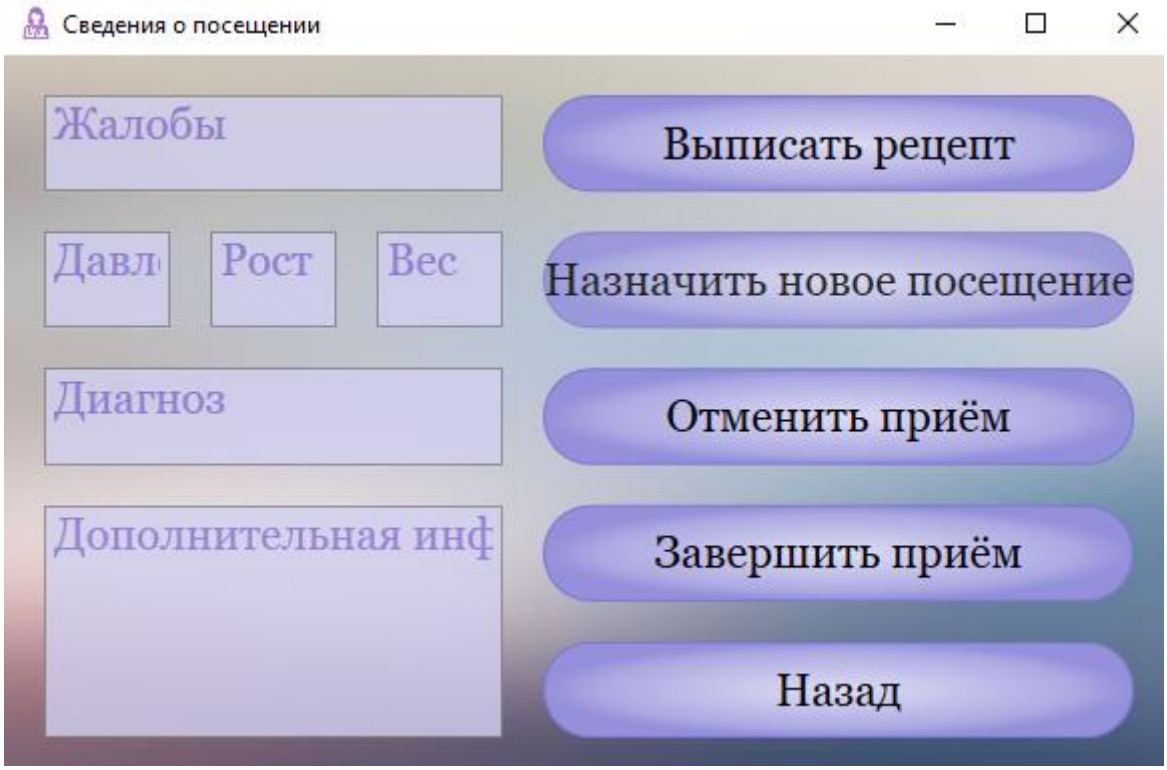


Рисунок 6.3.5 – Окно для записи информации о посещении

На данном окне присутствует возможность записи информации о посещении, кнопки для отмены и завершения приёма, перехода на прошлое окно. Также есть возможность выписать рецепт. При нажатии на «Выписать рецепт» открывается окно, в котором требуется указать информацию о рецепте и нажать «Ок». В таком случае откроется диалоговое окно для изменения настроек печати. Также есть возможность назначить новое посещение для выбранного пациента. Процесс записи пациента на посещение был подробно описан в главе 6.3.4.

После завершения либо отмены приёма терапевт возвращается на главное окно терапевта.

ЗАКЛЮЧЕНИЕ

В результате курсовой работы было разработано программное средство «Терапевт», которое выполняет следующие функции:

– имеет интуитивно понятное и простое управление;

– обращается к базе данных, успешно извлекает и вносит новые данные, осуществляет поиск по базе и автоматическую запись некоторых полей;

– имеет различный функционал в зависимости от учётной записи. Терапевт может вводить информацию о приёме, создавать записи новых пациентов и изменять существующие, просматривать статистику, назначать новые приёмы, изменять информацию в базе данных, выписывать рецепты. Пациент может записываться на новые приёмы, администратор может добавлять и изменять записи терапевтов;

– выглядит аккуратно, не имеет лишнюю информацию, адаптирует размеры и расположение элементов управления в зависимости от размера окна.

Таким образом, разработанное программное средство удовлетворяет всем требованиям технического задания, а также реализует несколько дополнительных функций.

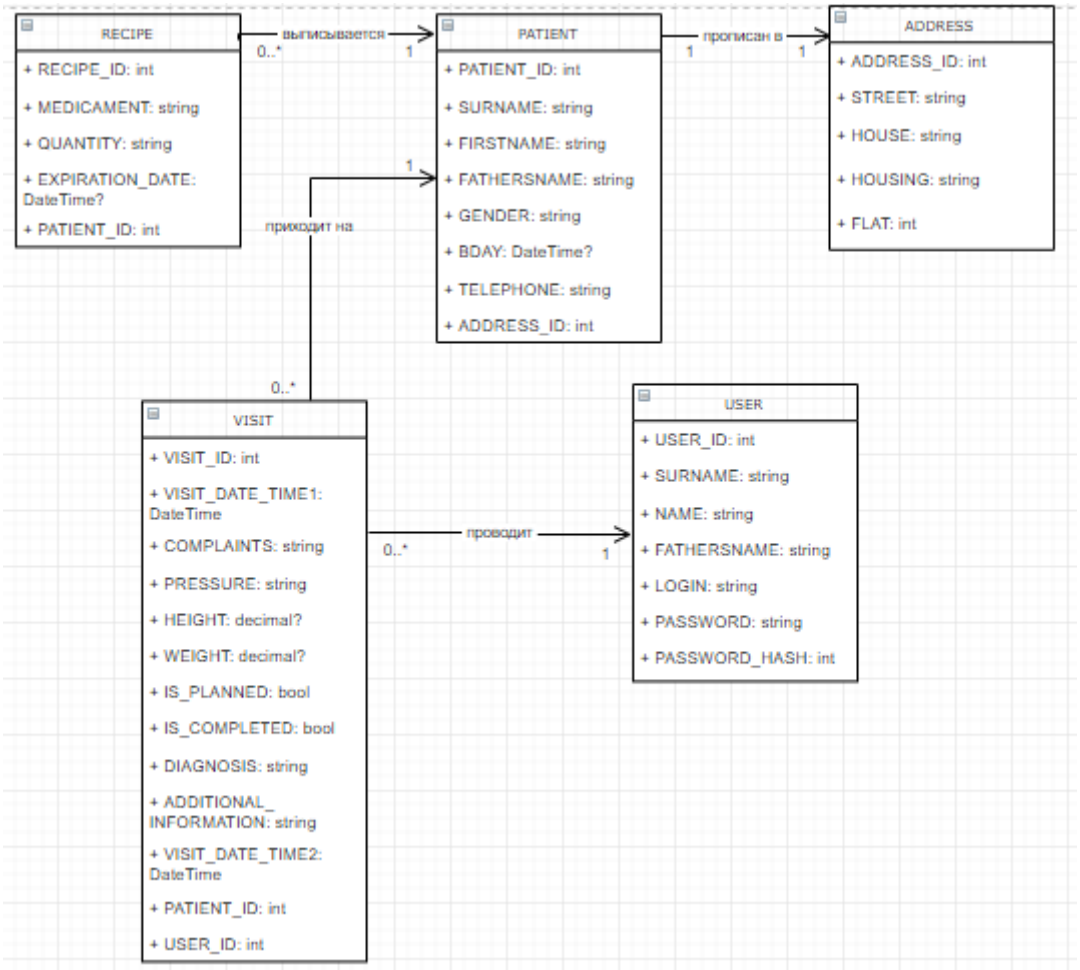
Данный программный продукт является завершённым и готов к использованию в медицинской отрасли. С учётом активной интеграции цифровых технологий во все сферы жизни, разработанное приложение является актуальным.

СПИСОК ЛИТЕРАТУРЫ

1. Mogcp [Электронный ресурс]. – Электронные данные. – Режим доступа: http://mogcp.by/clinic/appointments/appointments-clinic7.
2. Google Play [Электронный ресурс]. – Электронные данные. – Режим доступа: https://play.google.com/store/apps/details?id=ab.damumed&hl=ru.
3. MaxTarget [Электронный ресурс]. – Электронные данные. – Режим доступа: http://maxtarget.by/crm-sistemy/programma-uchet-patsientov.
4. Usu.kz [Электронный ресурс]. – Электронные данные. – Режим доступа: http://usu.kz/uchet_poseschenіy_v_poliklinike.php.
5. Medsoftpro [Электронный ресурс]. – Электронные данные. – Режим доступа: https://medsoftpro.ru/ambulatoria/description.html.
6. Park.by [Электронный ресурс]. – Электронные данные. – Режим доступа: http://www.park.by/post-834/.

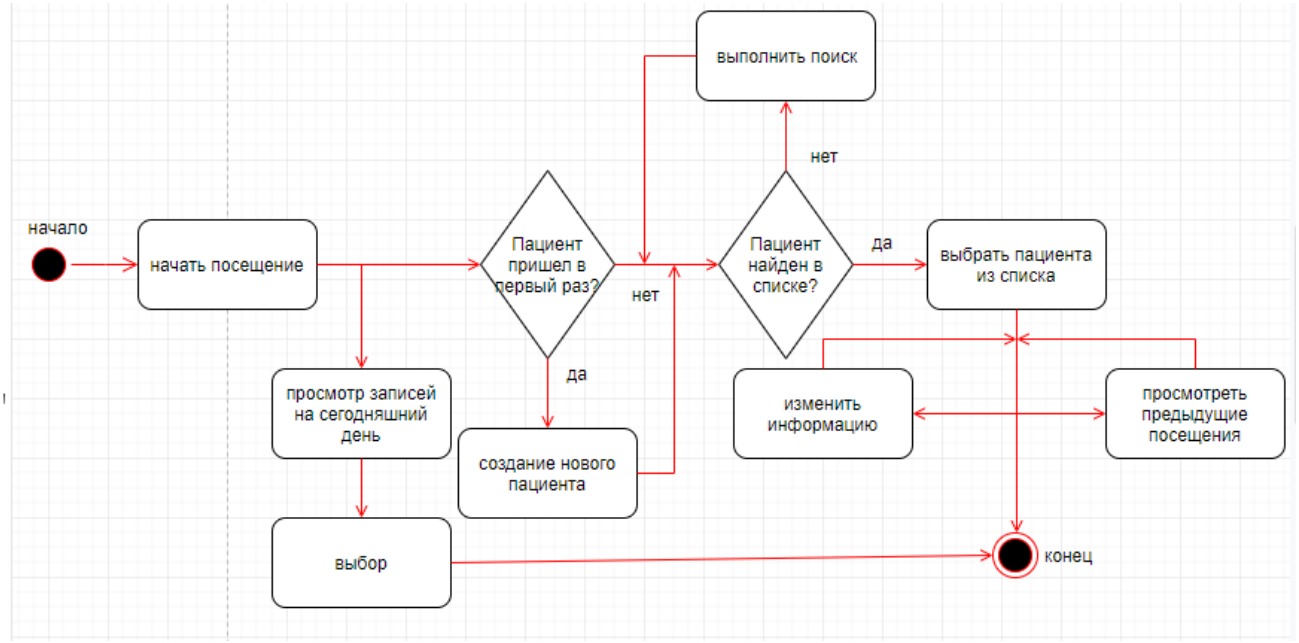
Приложение А

Диаграмма классов



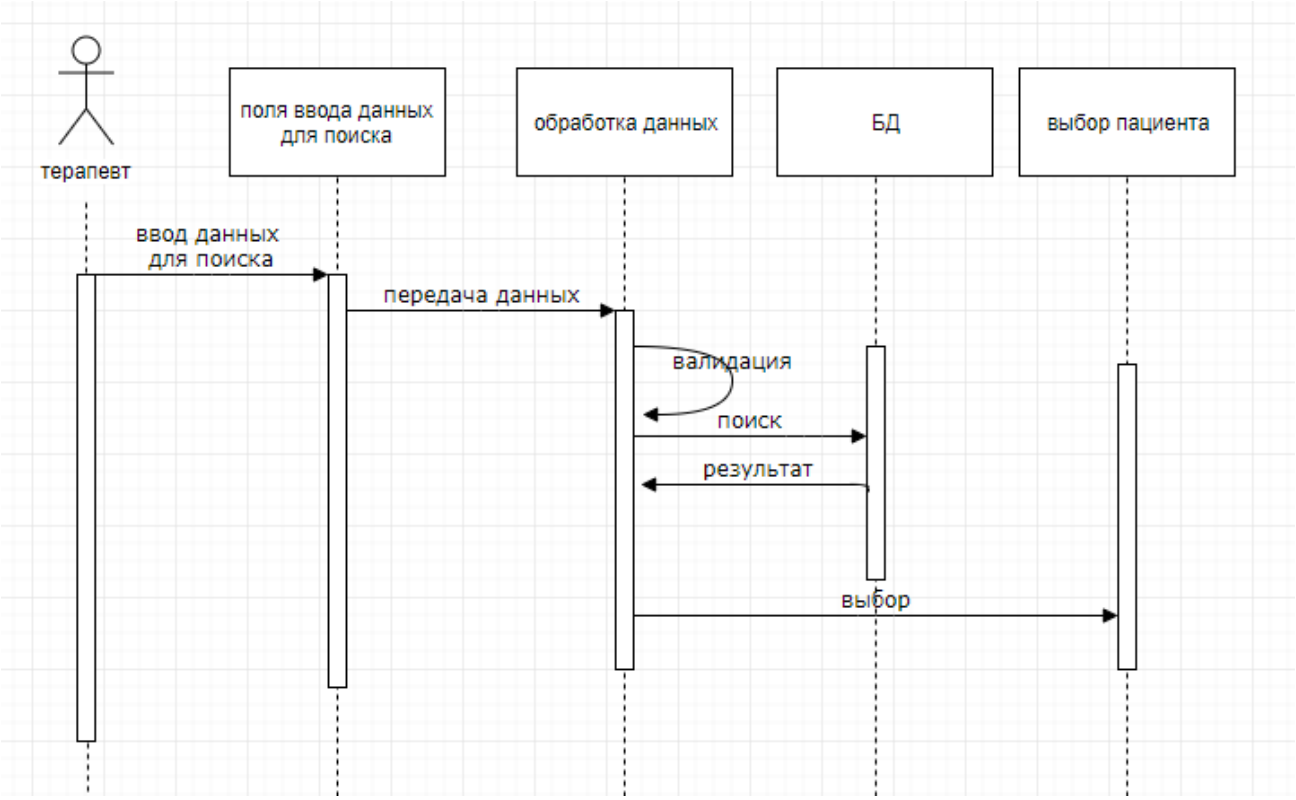
Приложение Б

Диаграмма деятельности



Приложение В

Диаграмма последовательности



Приложение Г

Скрипт создания базы данных

```
use master
go
create database KURSACH

use KURSACH
create table ADDRESS
(       ADDRESS_ID integer identity(1,1) constraint PK_ADDRESS primary key,
        STREET nvarchar(64) not null,
        HOUSE nvarchar(5) not null,
        HOUSING nvarchar(5),
        FLAT integer
)
create table PATIENT
(       PATIENT_ID integer identity(1,1) constraint PK_PATIENT primary key,
        SURNAME nvarchar(30) not null,
        FIRSTNAME nvarchar(30) not null,
        FATHERSNAME nvarchar(30),
        GENDER nchar(1) CHECK (GENDER in ('м', 'ж')),
        BDAY date,
        TELEPHONE nvarchar(15),
        ADDRESS_ID integer constraint FK_ADDRESS_PATIENT foreign key references ADDRESS(ADDRESS_ID)
)
create table USERS
(
        [USER_ID] integer identity(1,1) constraint PK_USER primary key,
        LOGIN nvarchar(15) not null,
        SURNAME nvarchar(20) not null,
        NAME nvarchar(20) not null,
        FATHERSNAME nvarchar(20) not null,
        PASSWORD_HASH integer not null,
        CHANGE nchar(1) CHECK (CHANGE in ('1', '2'))
)

insert into USERS
        values ('admin', 'Дубень', 'Полина', 'Васильевна', 1951755706, 1), -- пароль ‘1q1qadmin’
                ('pomazafa', 'Кричалов', 'Кирилл', 'Анатольевич',
```

```
2019917272, 2)

CREATE TABLE VISIT
(
	VISIT_ID integer  identity(1,1) constraint PK_VISIT primary key,
	PATIENT_ID integer constraint  FK_VISIT_PATIENT foreign key

references PATIENT(PATIENT_ID),
	VISIT_DATE_TIME1 datetime not null,
	COMPLAINTS nvarchar(300),
	PRESSURE nvarchar(10),
	HEIGHT decimal(4,1),
	WEIGHT decimal(5,2),
	IS_PLANNED bit not null,
	IS_COMPLETED bit not null,
	DIAGNOSIS nvarchar(max),
	ADDITIONAL_INFORMATION nvarchar(max),
	[USER_ID] integer constraint FK_VISIT_USER foreign key

		references USERS([USER_ID]),
	VISIT_DATE_TIME2 datetime default '01-01-1970'
)
 create table RECIPE
 (
	RECIPE_ID integer  identity(1,1) constraint PK_RECIPE  primary
key,
	MEDICAMENT nvarchar(30) not null,
	QUANTITY nvarchar(20) not null,
	EXPIRATION_DATE date,
	PATIENT_ID integer constraint FK_RECIPE_PATIENT foreign key

		references PATIENT(PATIENT_ID)
)
```

Приложение Д

Структура базы данных