Введение

В дипломном проекте рассматриваются основные концепции информационной системы документооборота и выделяются основные моменты, характерные для решения данной задачи. Описывается процесс реализации разработанных моделей с помощью языка программирования.

Прогресс достигнутый за последние несколько лет во всех аспектах вычислительной техники, включая теорию, технологию и приложения, привели к значительному расширению области применения компьютеров и росту числа их пользователей. Существенной частью современного общества являются разнообразные системы доступа и хранения информации, которые являются неотъемлемой составляющей современного научно-технического прогресса. Существует много веских причин перевода существующей информации на компьютерную основу, так как более быстрая обработка данных и централизация их хранения с использованием клиент/серверных технологий позволяют сберечь значительные средства, а главное и время для получения необходимой информации, а также упрощает доступ и ведениях.

В любой организации, как большой, так и маленькой, возникает проблема такой организации управления данными, которая обеспечила бы наиболее эффективную работу. Некоторые организации используют для этого шкафы с папками, но большинство предпочитают компьютеризированные СУБД, позволяющие эффективно хранить, извлекать информацию и управлять большими объемами данных. Современные СУБД – многопользовательские системы управления базой данных, которые специализируется на управлении массивом информации одним или множеством одновременно работающих пользователей.

В частности, при изучении проблем, с которыми сталкивается комитет судебных экспертиз, выяснилось, что для составления отчетов в текстовом редакторе Microsoft Word ведущему специалисту приходится составлять формы отчетности и заполнять их данными с клавиатуры, что является очень трудоемким занятием, отнимающим значительную часть рабочего и свободного времени. При данном методе ведения документации велика вероятность появления ошибок при наборе. На жестком диске компьютера постепенно накапливается несистематизированная или систематизированная со слабыми связями информация в виде набора файлов и каталогов, где для поиска нужной информации необходимо открывать каждый третий файл и постепенно собирать общую информацию по критерию. Это рутинное занятие не привлекает конкретного работника, на которого возлагается эта обязанность вдобавок к основной расчетной деятельности, тем более за оклад, который не предполагает выполнение чрезмерной бумажной работы. Поэтому желание руководства предприятия по созданию компьютерной системы для автоматизации создания выходной документации, то есть отчетов, актов сверки и так далее вполне объяснимо. Это и оперативность при работе с отчетами.

От пользователя достаточно иметь навыки работы с операционной системой Microsoft Windows и своевременно вести учет записей в базе данных программного продукта, о котором идет речь. Ведение учета состоит в заполнении информации, если информация в базе данных отсутствует.

Составлением всех отчетных документов занимается один человек, который в силу человеческого фактора может сделать ошибки при расчетах и заполнении документов. Хотя для этого и используется персональный компьютер, вероятность ошибок существует потому, что решается эта проблема путем набора документов с клавиатуры с помощью текстового редактора Microsoft Word, что лишь повышает качество документа, но никак не способствует его устойчивости к появлениям ошибок при наборе. В силу этих причин появляется необходимость в большем объеме использовать ресурсы компьютера, то есть не только набор и распечатка текста, а автоматическое ведение учета. Программа, которая взяла всю рутинную работу на себя, то есть производила калькуляцию и подведение итогов, а также вывод отчетов, была бы не лишней для данного представительства.

Целью данной дипломного проекта является создание программного обеспечения для автоматизации процесса создания выходной документации для судебно-медицинской экспертизы.

Для создания программного продукта необходимо решить следующие задачи:

1. изучить систему ведения учета;

2. описать эту систему в базе данных, то есть создать базу данных на основе этой системы;

3. установить точки входа и выхода информации для базы данных, то есть, какую информацию необходимо указывать пользователю, что нужно отображать на экране или в отчетах, а что использовать для служебного назначения, например, для управления записями, то есть для создания запросов;

4. выявить недостатки в структуре базы данных и сделать корректировку соответственно требованиям пользователя;

5. разработать программный продукт для управления базой данных, то есть программное средство, которое будет осуществлять добавление, редактирование и удаление записей, а также выборку данных по определенным критериям и оформление результата выборки в отчеты;

6. предоставить разработанный программный продукт для ознакомления с ним пользователю, определение недостатков и принятие к доработке замечаний.

Объектом исследования данного дипломного проекта является учет алкогольное, наркотические и полизависимых людей РБ. Предметом исследования является автоматизация процесса создания выходной документации для «судебно-медицинских экспертиз».

1. Анализ существующих решений по теме проекта и постановка задачи

1.1 Анализ существующих решений по теме проекта

1.1.1 Database Microsoft Access

Программа реализована на Microsoft Access

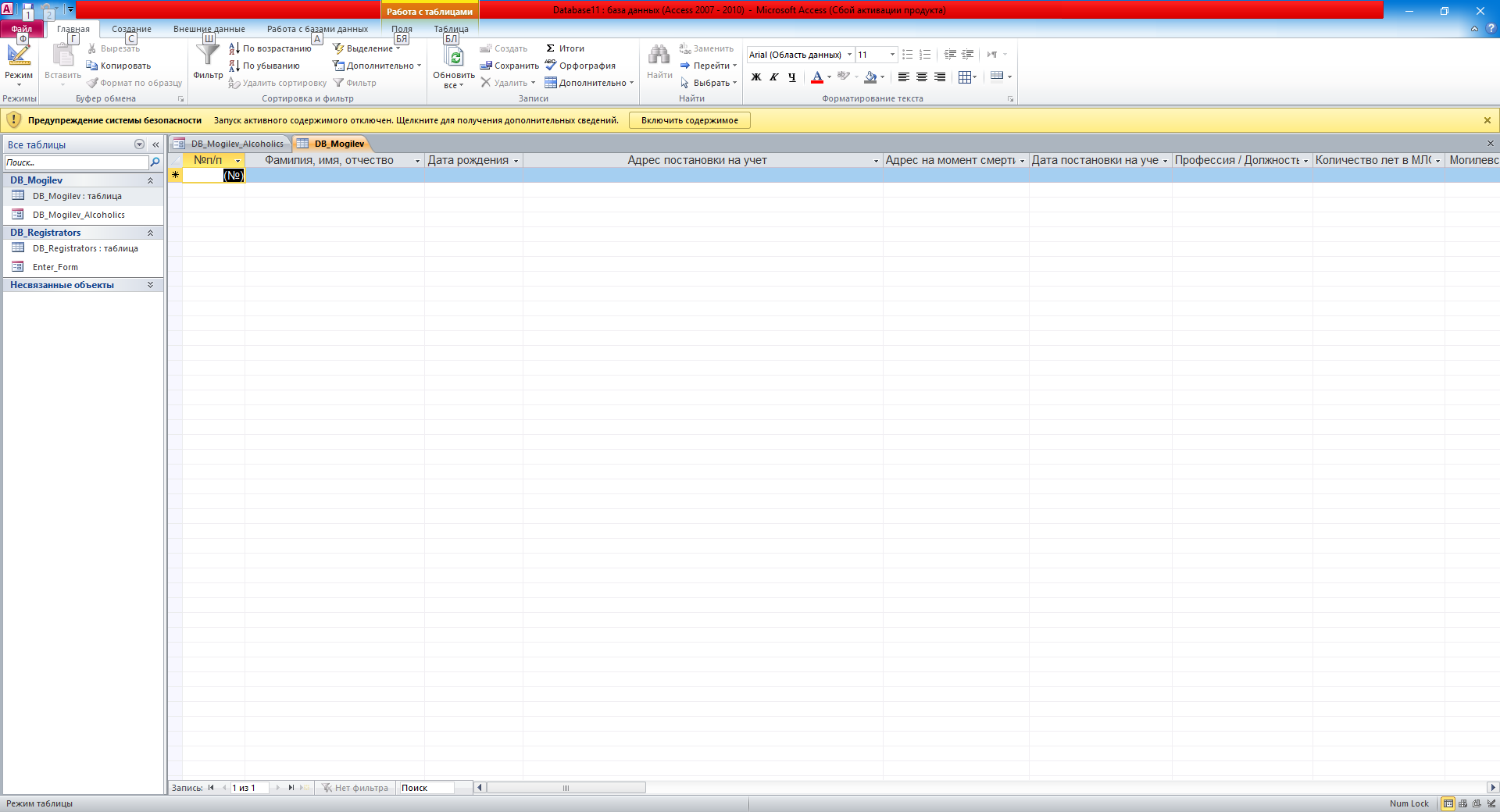


Рисунок 1.1.1.1 Database Microsoft Access

Создав новую базу данных. Вы можете заняться её заполнением вручную либо наполнить, копируя необходимую информацию из интернета. Обратите внимания, что каждая внесённая вами часть информации должна быть размещена в отдельной колонке, а каждая запись иметь личную строку. Кстати, колонки можно переименовать, чтобы лучше ориентироваться в их содержимом.

Если же вся необходимая вам информация находится в другой программе или источнике, программа позволяет настроить импортирование данных.

Все настройки импортирования находятся в отдельно вынесенной в панель управления вкладке, которая называется «Внешние данные». Здесь, в области «Импорт и связи», перечислены доступные форматы, среди которых документы Excel, Access, текстовые и XML-файлы. Выбрав необходимый формат, с которого будет перенесена информация, вам потребуется указать путь к месторасположению файла. Если же она размещена на каком-либо сервере, программа потребует от вас ввести адрес сервера. По мере импорта столкнётесь с различными настройками, которые предназначены для корректного переноса данных в Access.

Запрос является действием в программе, благодаря которому пользователь может редактировать или вносить информацию в базу данных. Фактически запросы делятся на 2 вида:

Выборный запросы, благодаря которым программа достаёт определённую информацию и делает по ней расчёты.

Запросы действия, добавляющие информацию в базу либо удаляющие её.

Выбрав «Мастер запросов» во вкладке «Создание», программа проведёт процесс создания [определённого типа запроса](http://nastroyvse.ru/programs/review/kak-sozdat-zapros-access.html). Следуйте инструкциям.

Запросы могут значительно помочь вам упорядочить данные и всегда обращаться к конкретной информации.

Например, вы можете создать выборный запрос по определённым параметрам. Если вы хотите увидеть информацию по конкретной дате или дню таблицы «День» за весь промежуток времени, вы можете настроить подобный запрос. Выберите пункт «Конструктор запросов», а в нём нужную вам таблицу. По умолчанию запрос будет выборным, это становится понятным, если глянуть на панель инструментов с выделенной там кнопкой «Выборка». Для того чтобы программа искала именно ту дату или день, который вам нужен, найдите строку «Условие отбора» и введите туда фразу [какой день?]. Запомните, запрос должен быть помещён в квадратные дужки [] и заканчиваться на знак вопроса либо на двоеточие.

Это лишь один из вариантов использования запросов. На самом деле, с их помощью также можно создавать новые таблицы, отбирать данные по критериям и т. д.

Настройка и использование форм

Благодаря [применению форм](http://nastroyvse.ru/programs/review/sozdanie-form-microsoft-access.html), пользователь сможет легко просматривать сведения по каждому полю и переключаться между существующими записями. При длительных вводах информации использование форм упрощает работу с данными.

Откройте вкладку «Создание» и найдите пункт «Форма», нажав на который появится стандартная форма, основанная на данных вашей таблицы. Появившиеся поля с информацией подвергаются всяческим изменениям, включая высоту, ширину и т. д. Обратите внимание, что, если в приведённой таблице присутствуют взаимосвязи, вы увидите их и сможете перенастроить в этом же окне. В нижней части программы вы увидите стрелки, которые позволят последовательно открывать каждый столбец вашей таблицы либо же сразу переместиться на первый и последний. Теперь каждая из них является отдельной записью, поля которой вы можете настроить, кликнув мышью на кнопку «Добавить поля». Изменённые и внесённые таким образом сведения отобразятся в таблице и во всех присоединённых к ней таблицах. После настройки форма, необходимо сохранить её, нажав комбинацию клавиш «Ctrl+S».

Создание отчёта

Основным предназначением отчётов является предоставлению пользователю общей сводки по таблице. [Отчёт можно создать](http://nastroyvse.ru/programs/review/kak-sozdat-otcet-access.html) абсолютно любой, в зависимости от данных.

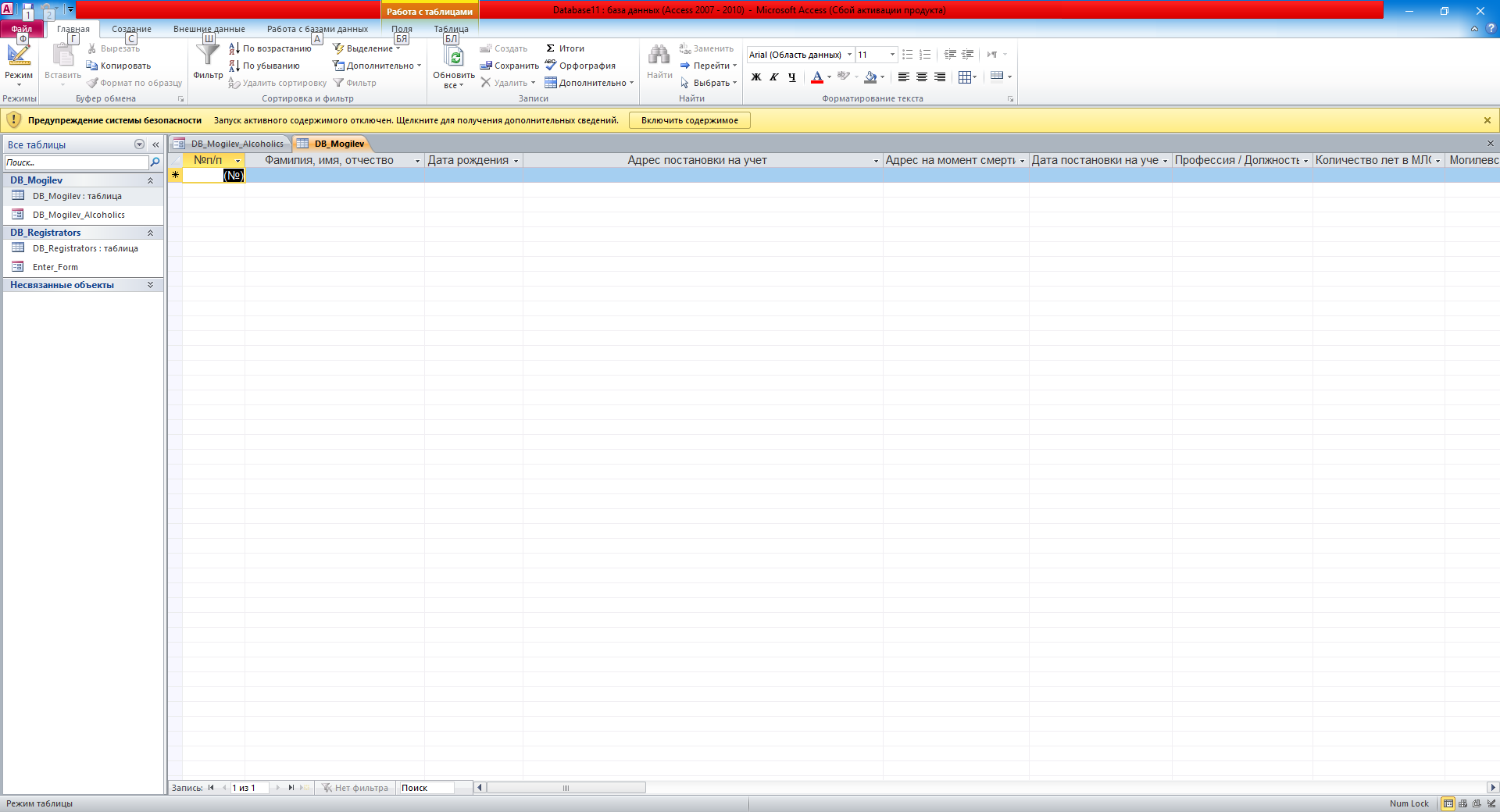


Рисунок 1.1.1.2 Формирование отчета

Программа даёт выбрать тип отчёта, предоставив несколько на выбор:

Отчёт — будет создан автоотчёт, использующий всю предоставленную в таблице информацию, однако, данные не будут сгруппированы.

Пустой отчёт — не заполненная форма, данные для которой вы сможете выбрать сами из необходимых полей.

Мастер отчётов — поможет вам пройти процесс создания отчёта и проведёт группирование и форматирование данных.

В пустой отчёт можно добавить, удалить или редактировать поля, наполняя их необходимой информацией, создавать специальные группы, которые помогут отделить определённые данные от остальных и многое другое.

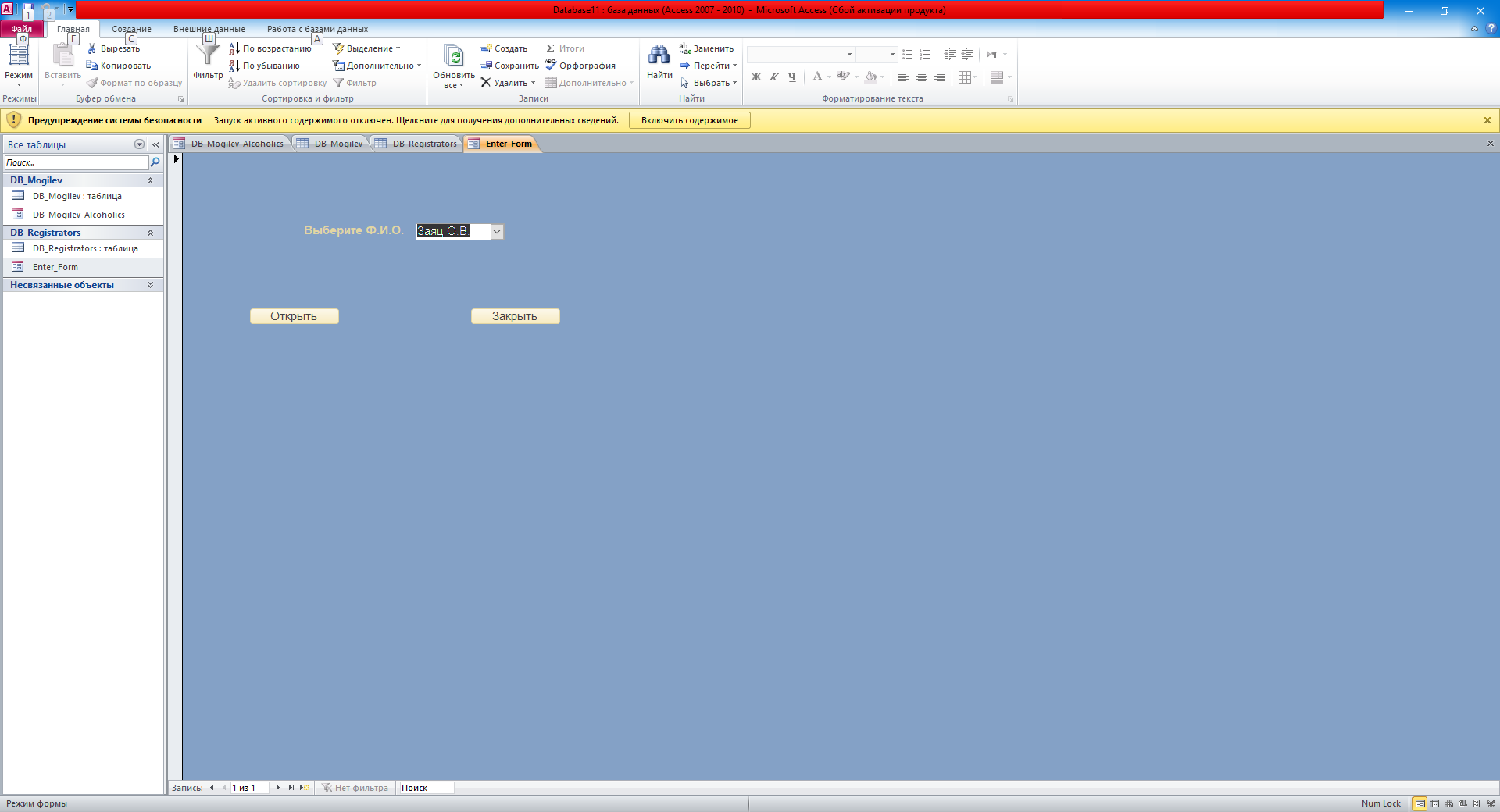
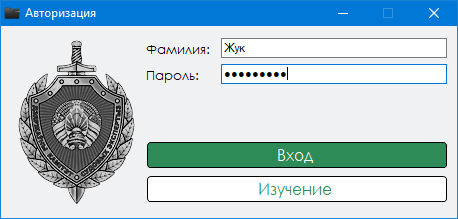


Рисунок 1.1.1.3 Окно загрузки формы отчета

Недостатки этой программы заключается в том, что всю информацию надо вводить все вручную.



1.2 Forensic Medical Expertise. Форма ввода логина и пароля.

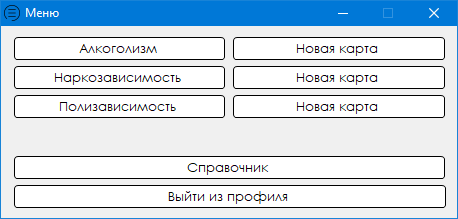


Рисунок 1.1.2.1 Главное окно Forensic Medical Expertise.

Через главное окно рис 1.1.2.1 происходит выбор на формирование отчетов или добавление нового персонажа в БД. Также тут размещена кнопка справочника, информация о программе, выход из профиля и выход из программы.

В окне новая карта (рис.1.1.2.2 – рис 1.1.2.4), вносится информация по специальному критерию (в зависимости от выбранного главного критерия). Заполнение форм реализуется через TextBox, ComboBox и Radiobatton.

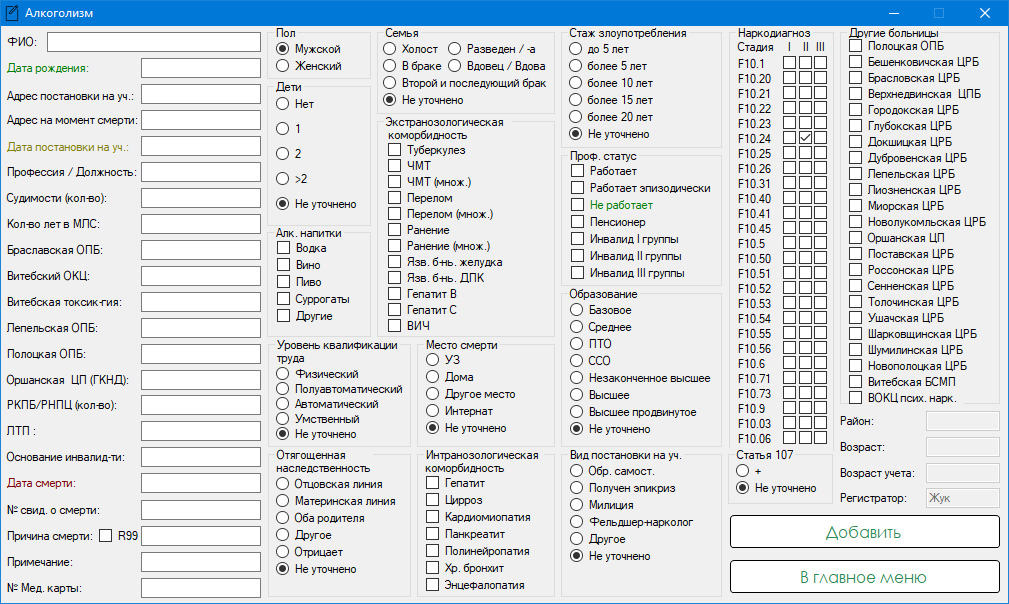


Рисунок 1.1.2.2 Окно новая карта Forensic Medical Expertise

После того как форма будет заполнена форма она добавляется в DataGrid (рис 1.1.2.5). После этого информация записывается в Excel.

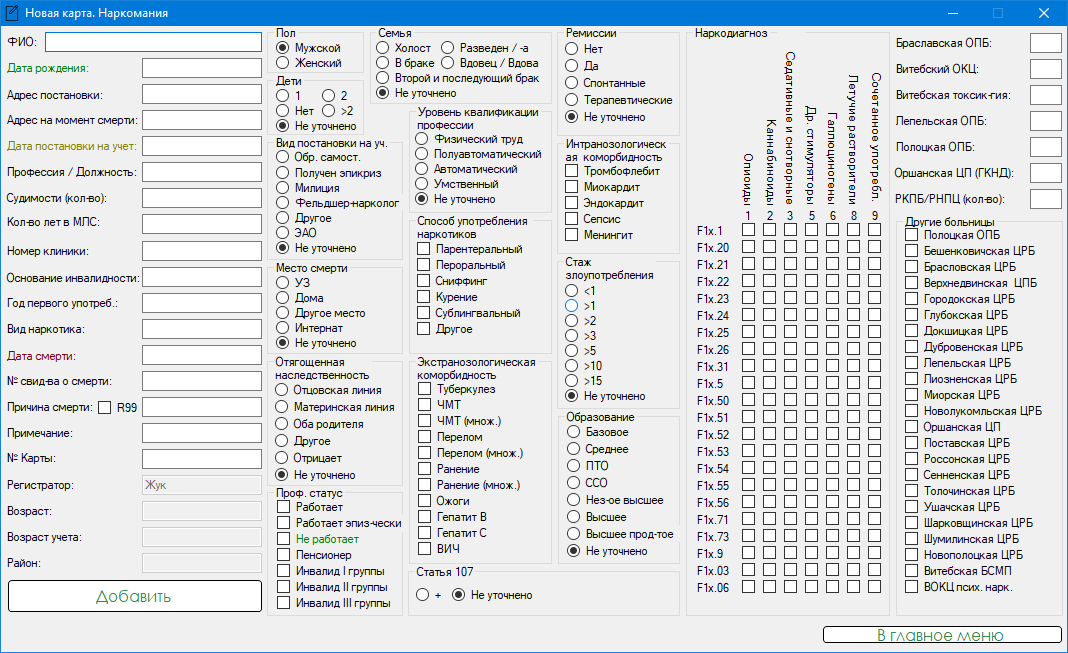


Рисунок 1.1.2.3 Окно новая карта Forensic Medical Expertise.

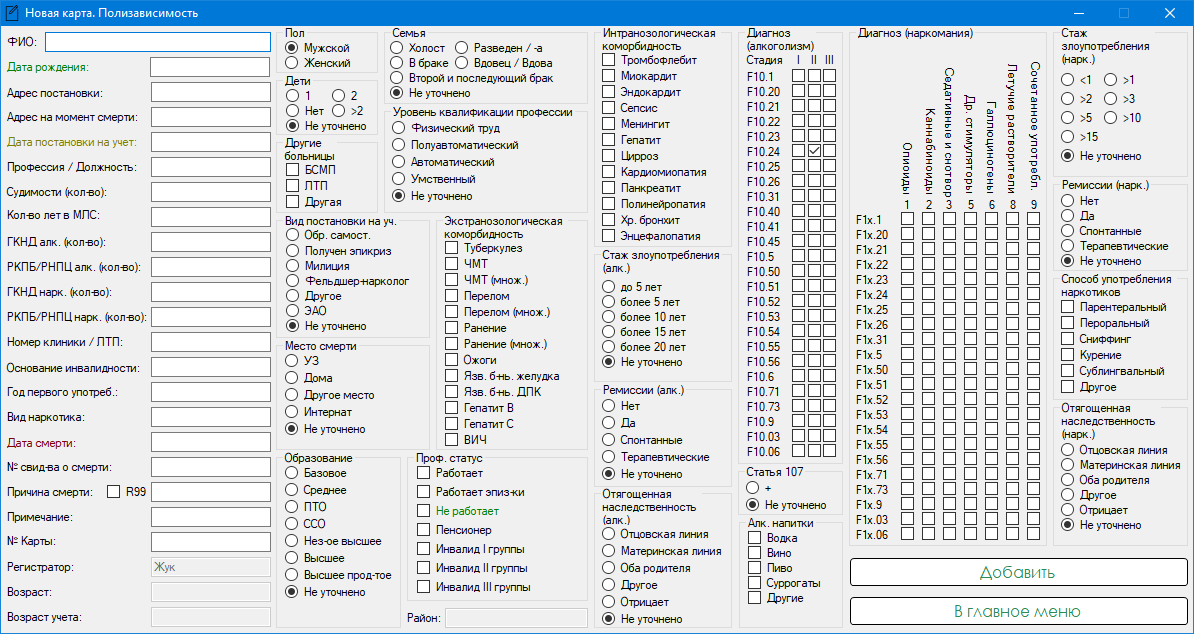


Рисунок 1.1.2.4 4 Окно новая карта Forensic Medical Expertise.

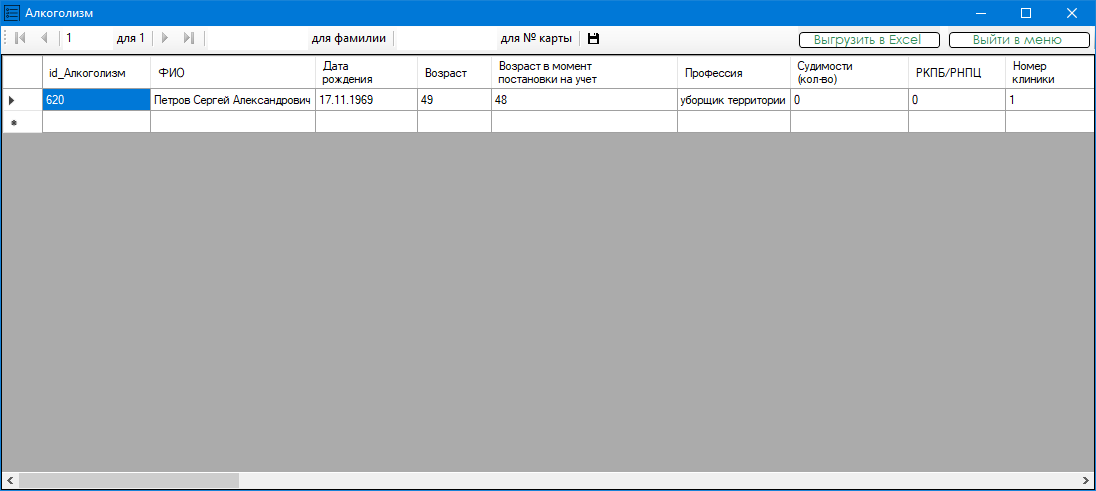


Рисунок 1.1.2.5 Окно выборки информации Forensic Medical Expertise

Данная программа предназначена только на один компьютер, при работе на несколько компьютерах, приходится копировать информацию на флэшку. После копировать информацию на один компьютер. Там объединять информацию, затем производить анализ данных и формировать отчет по анализу информации.

Недостаток данной программы:

- Программа не использует не одну из базы данных, она все сохраняет сразу в Excel файле.

- Программа не содержит все требование для формирования карты пациента.

- Программа не формирует не каких статистических методов.

2 Методы и модели, положенные в основу проекта

2.1 Постановка задачи

Предназначен программный продукт для добавления, обработки и хранения информации о поставленных на учет в различных областях РБ, алкоголиков, наркоманов и полизависимых. Язык программирования программного продукта - С#.

Бизнес логика программного продукта должна быть абстрагирована от графического представления, для возможности интеграции в существующие программные продукты комплексного управления комитета судебных экспертиз. Графическое представление должно иметь визуально-привлекательные возможности взаимодействия с пользователей в случае использование программного продукта в качестве самостоятельного автономного приложения. Наиболее полно данным требованиям отвечает система для построения клиентских приложений Windows –WPF, графическая подсистема некоторая использует язык XAML.

Для документирования вариантов использования и действующих лиц, а также для создания диаграмм, показывающих взаимодействие между ними использовать объектно-ориентированное средство проектирование RATIONAL ROSE.

Для создания и взаимодействия с базой данных использовать специальную объектно-ориентированную технологию на базе Microsoft.Net Framework – Microsoft Framework Code First.

Для управление базой данных использовать систему управление реляционными базами данных – Microsoft SQL Server 2016.

Программный продукт должен реализовать следующие функции:

- открытие дочерних окон редактирования данных и формирование отчетов с главной страницы программного продукта;

- добавление и удаление информации по отношению лиц, поставленных на учет.

- Выборка данных с отображение на экран по виду зависимости;

- Выборка данных с отображение на экран по областным центрам;

- Выборка данных с отображение на экран по половому признаку;

- Выборка данных с отображение на экран по дате постановке на учет;

- Выборка данных с отображение на экран по дате смерти;

- Выборка данных с отображение на экран по семейному статусу;

- Выборка данных с отображение на экран по количеству детей;

- Выборка данных с отображение на экран по составу семьи;

- Выборка данных с отображение на экран по уровню образования;

- Выборка данных с отображение на экран по году постановки на учет;

- Выборка данных с отображение на экран по году смерти;

- Экспортирования результатов выборки данных в формат Excel

- открытия Excel по выбору пользователя для редактирования из окон выборки данных

Таким образом отдельные инструменты разработки и основные требования к программной системе. Использования системы WPF позволит максимально абстрагировать интерфейс, предоставляемый пользователю от логики приложения, что позволит максимально быстро интегрировать данную программную систему в уже существующие на предприятии.

2.3 Модели представление систем и их описание

Модель представления системы может рассматривались с *различных точек* зрения (представлений). Здесь наиболее интересным и информативным представляется представлен неиспользования (Use Case View) это описание поведения системы с точки зрения внешних по отношению к системе действующих лиц. Варианты использования системы рассмотрим на основании диаграмм вариантов использования (use – case diagram) на которых изображены отношения, существующие между актерами (actor) и вариантами использования системы (use -case).

В нашем случае актерами являются пользователи, использующие программную систему под определенными ролями, которые в свою очередь определены требованиями. На рисунке 2.3.1 показаны актеры программной системы.

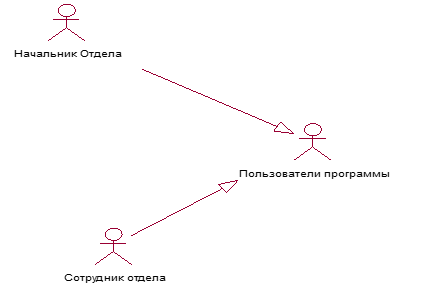


Рисунок 2.3.1 - Сотрудники предприятия и их роли при взаимодействии программной системой

В соответствии с назначенной ролью каждый актер выполняет определенный набор функций в программной системе. На рисунке 2.3.2 приведены диаграммы вариантов использования для каждой из ролей актеров

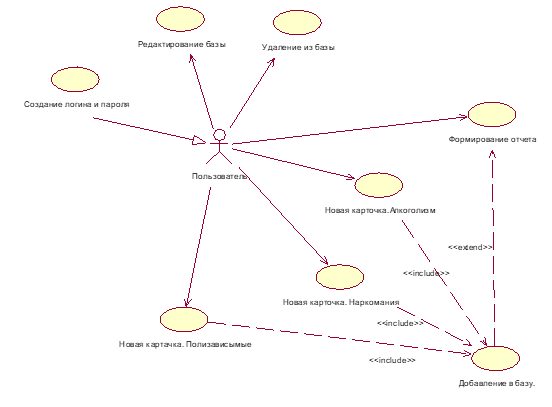


Рисунок 2.3.2 - Use-case диаграмма для роли пользователь

3 Разработка проекта программного обеспечения (формирование функциональных требований, разработка структуры, разработка схемы данных реляционной модели)

Анализируя предметную область можно выделить ряд сущностей, которые будут кандидатами для формирования классов программной системы.

В предметной области можно выделить три основные подобласти:

Алкогольно зависимые люди, наркозависимые люди, поли зависимые люди.

Все три области относятся к одному объекту - комитету судебной экспертизе.

В атрибутах каждой сущности присутствуют одинаковые атрибуты, которые можно объединить в отдельные абстрактные классы.

Описание классов и архитектуры программной системы

На основании выявленных сущностей их атрибутов и отношений между ними составим описание классов программной системы. Для лучшего понимания архитектуры программной системы составим диаграмму классов.

Классы программной системы для алкозависымых, наркозависымых, полизависимых будут рассмотрены в рамках дипломного проекта.

Архитектура программной системы представлена тремя слоями и является многослойной. Многослойная архитектура призвана разделить функциональные области приложения на группы. В нашем случае выделим три группы, представляющие собой слои пирамиды, каждый слой которой знает о существовании нижележащего, но не ниже. Это позволит строго разделить функциональность приложения, что в свою очередь обеспечит гибкость и простоту обслуживания. Связь между слоями осуществляется реализацией интерфейсов в виде методов доступа к сервисам нижележащих слоев. На рисунке 2.2.1 представлена архитектура приложения.

Верхний слой представляет собой слой пользовательского интерфейса (UIL - User Interface Layer). В нашем случае он представлен страницами WPF приложения. Основная задача этого уровня предоставлять эргономичный интерфейс пользователю в соответствии с заявленным функционалом.

Средний слой или слой бизнес-логики (BLL — Business Logic Layer) содержит классы, объекты которых непосредственно отображаются на слой пользовательского интерфейса, сервисы для работы с нижним уровнем данных,

I основную логику проверки валидности вводимых пользователем данных.

Нижний слой или слой доступа к данным (DAL - Data Access Layer) отвечает за взаимодействие с базой данных, а в случае с технологией «Code First» еще и за формирование ее архитектуры. Архитектура базы данных в нашем случае определяется классами данных определенных на этом уровне. Основные методы по взаимодействию с базой данных определены в интерфейсе

Для начала разберем классы уровня данных, которые непосредственно участвуют в формировании базы данных и соответствующие им классы уровня бизнес логики, непосредственно участвующие в реализации методов CRUD на стороне пользовательского интерфейса.

Класс Alco, соответствует сущности алкоголизм. Для формирования связи один ко многим содержит в конструкторе типизированные коллекции связанных классов, а также навигационные свойства по каждой связанной таблице.

Для класса Alco создан класс AlcogolismPerson, реализующий основные функции работы с базой данных.

На уровне бизнес логики классу Alco соответствует класс AlcoViewModel. В данном классе определяем для каждого свойства класса правила валидации. Сделано это для проверки введенных данных на уровне бизнес логики и отправки в базу данных валидных данных, если определить данные правила на уровне данных, то пользователь об ошибке узнает после ввода всей формы и попытке сохранения данных. Определение правил валидации на классе представления позволяет еще на стадии ввода *значения* конкретного свойства идентифицировать ошибку и оповестить пользователя об этом, в свойствах типа string при помощи регулярных выражений определены требования по составу символов, формирующих строку, их минимальному и максимальному содержанию в строке. По числовым значениям устанавливаем ограничения по минимальному и максимальному значению и возможности ввести отрицательные значения. В свойствах класса отражены коллекции зависимых классов.

Для формирования отчета на уровне бизнес *логики создан класс* ExcelDataServis. *Главной задачей* методов класса ExcelDataServis, является считывание из БД информации, согласно критериям, заданным в окне формирования отчета, затем записи информации в Excel файл, который в дальнейшем будет передан в отдел аналитики.

Для взаимодействия с базой данных на уровне бизнес *логики создан класс* Entity. *Главной задачей* методов класса Entity, является считывание и запись информации из БД.



Рисунок 3.1 – Архитектура программного продукта

Для начала берем классы уровня данных, которых непосредственно участвуют в формировании базы данных и соответствующие им классы уровня бизнес логики, непосредственно участвующие в организации методов CRUD на построение пользовательского интерфейса.

Формирование базы данных осуществляется с использованием технологии «Code First». Для переопределения свойств по умолчанию при формировании таблиц в БД будем использовать специальный атрибуты Атрибут [Requiered] указывает на то что данное свойство и соответствующие поле в базе данных обязательно должны содержать значение что соответствует NOT NULL.

Атрибуты для валидации вводимых данных определим по умолчанию. Это даст возможность определять правила валидации на уровне свойств классов бизнес логики.

Каждый слой приложения имеет свои наборы классов для реализации его функции. Слои взаимодействуют при помощи, специализированных сервисов, которые содержат методы для взаимодействия с нижележащими слоями

Таким образом в процессе анализа предметной области выявлены основные сущности программной системы и определены их атрибуты.

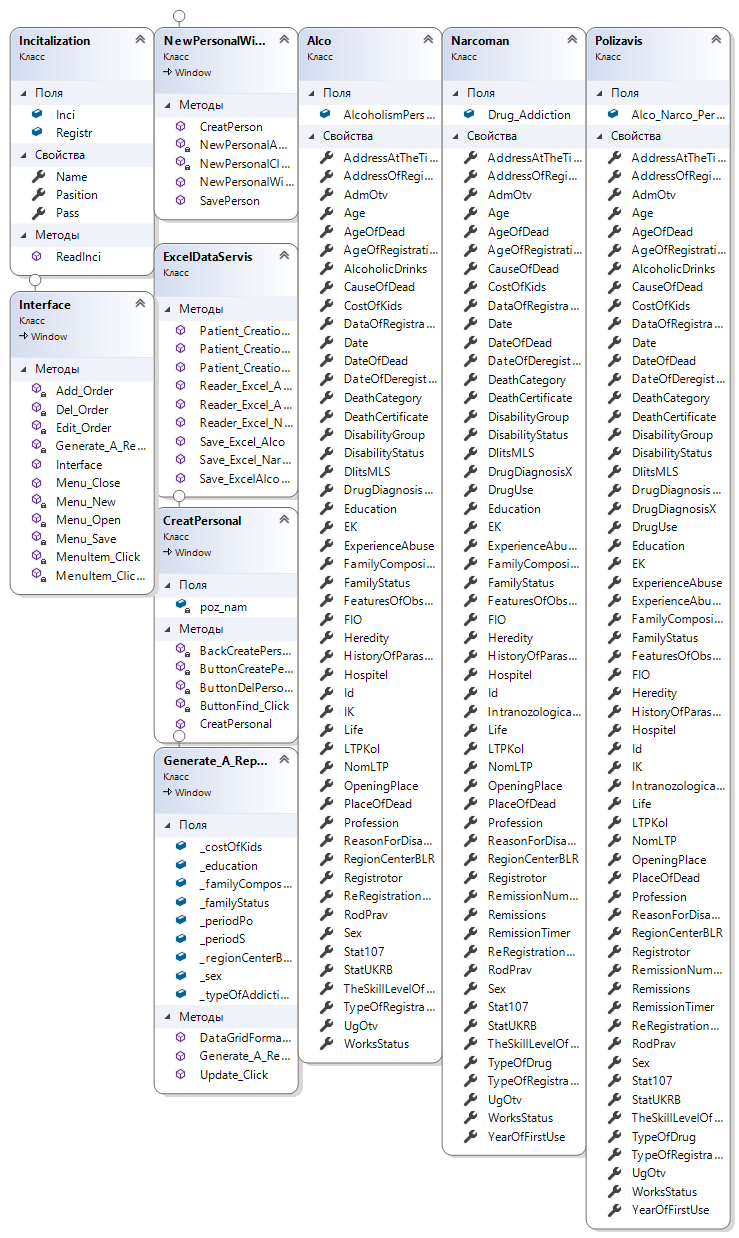
На рисунке 3.1 представлена диаграмма классов.

Рисунок 3.1 – Диаграмма классов

Отобразим описание используемых классов в программе в Таблице 3.1.1.

Таблица 3.1 – Описание классов программы

|  |  |
| --- | --- |
| **MainWindow** | Вызывает окно инициализации и входа в программу, проверяет и дает доступ к определённым параметрами |
| **Interface** | Вызывает главное окно, где происходит выбор создание пациента по критерию его зависимости или сформировать отчет, добавить персонал, удалить или редактировать. |
| **MainDataWindow** | Вызывает окно где вносятся данные по алкозависимым людям и добавляет его в базу |
| **MainDataWindow2** | Вызывает окно где вносятся данные по наркозависимым людям и добавляет его в базу |
| **MainDataWindow3** | Вызывает окно где вносятся данные по полизависимым людям и добавляет его в базу |
| **ExcelLog** | Создает файл для записи ошибки при работе программы |
| **ExcelDataServis** | Содержит логику на сохранение и считывание информации с Excel файла |
| **Generate\_A\_Report** | Вызывает окно для формирования отчета, содержит логику на формирования отчета и производит сохранения информации в Excel файл |
| **NewPersonalWindow** | Позволяет добавить персонал для допуска к работе с файлом |
| **Alco** | Содержит информацию для создания профиля алкозависимого человека |
| **Incitalization** | Содержит информацию для создания профиля персонала |

Продолжение таблицы 3.1

|  |  |
| --- | --- |
| **Narcoman** | Содержит информацию для создания профиля наркозависимого человека |
| **Polizavis** | Содержит информацию для создания профиля полизависимого человека |

4 Разработка и реализация программного кода

Рассмотрим несколько алгоритмов реализации бизнес-логики приложения, а именно алгоритм входа в систему пользователем.

На рисунке 4.1 представлена форма в хода в систему. Форма на поделена на три части. В левой части размещен герб судебно – медицинской экспертизы РБ. В правом верхнем углу расположены два TextBox. Один для ввода ФИО и второй для ввода пароля. Также в правом нижнем углу расположены две клавиши. Первая для входа систему. Клавиша не активна пока не будут введены данные логина. Вторая для регистрации нового пользователя.

Алгоритм входа в систему пользователем представлен на рисунке 4.2

В случае если пользователь не ввел логин и пароль, ровно как ввод содержит ошибки выводится сообщение о необходимости скорректировать данные см рисунок 4.3. При корректном вводе данные проверяются на соответствие существующих в базе данных пользователей и при нахождении соответствия, пользователь входит в систему. На рисунке 4.1 отображена система входа в программу.

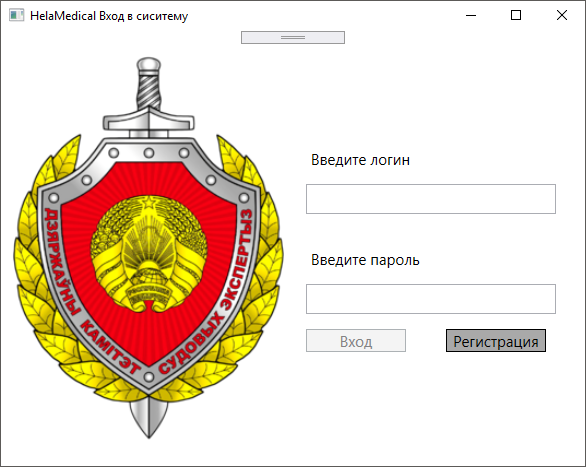


Рисунок 4.1 – Вход систему



Рисунок 4.2 Алгоритм входа в программу

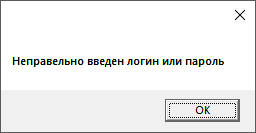


Рисунок 4.3 – отказ на вход в систему

Для регистрации нового сотрудника используется клавиша регистрация. При нажатии клавиши отрывается окно регистрации нового сотрудника рисунок 4.4.

Сотрудник должен ввести свою ФИО, ввести должность, и дважды ввести пароль.

Когда сотрудник заполняет все поля происходит сверка имени в БД и соответствии пароля. Если сверка имени нашла такого сотрудника в БД, то программа выдает ошибку «Такой пользователь уже существует», см рисунок 4.5.

При не заполненном одном из полей регистрации, программа также выводит ошибку и выделяет какой из полей не заполнено. Если все поля не заполнены, то программа также выдаст окно ошибки смотрите рисунок 4.6 – 4.7.

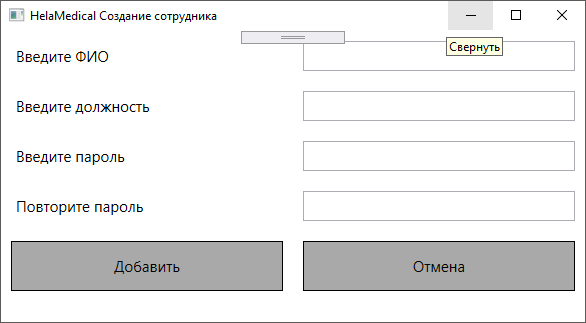


Рисунок 4.4 Окно регистрации сотрудника

Если проверка прошла успешно, то окно закрывается, и пользователь может войти в систему. Алгоритм представлен в приложение А.

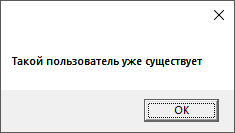


Рисунок 4.5 Окно ошибки при регистрации одинаковых пользователей

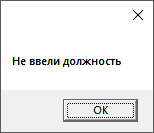


Рисунок 4.6 Окно ошибки, при не заполненном поле должность

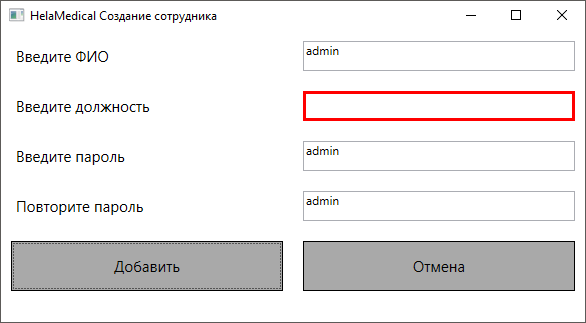
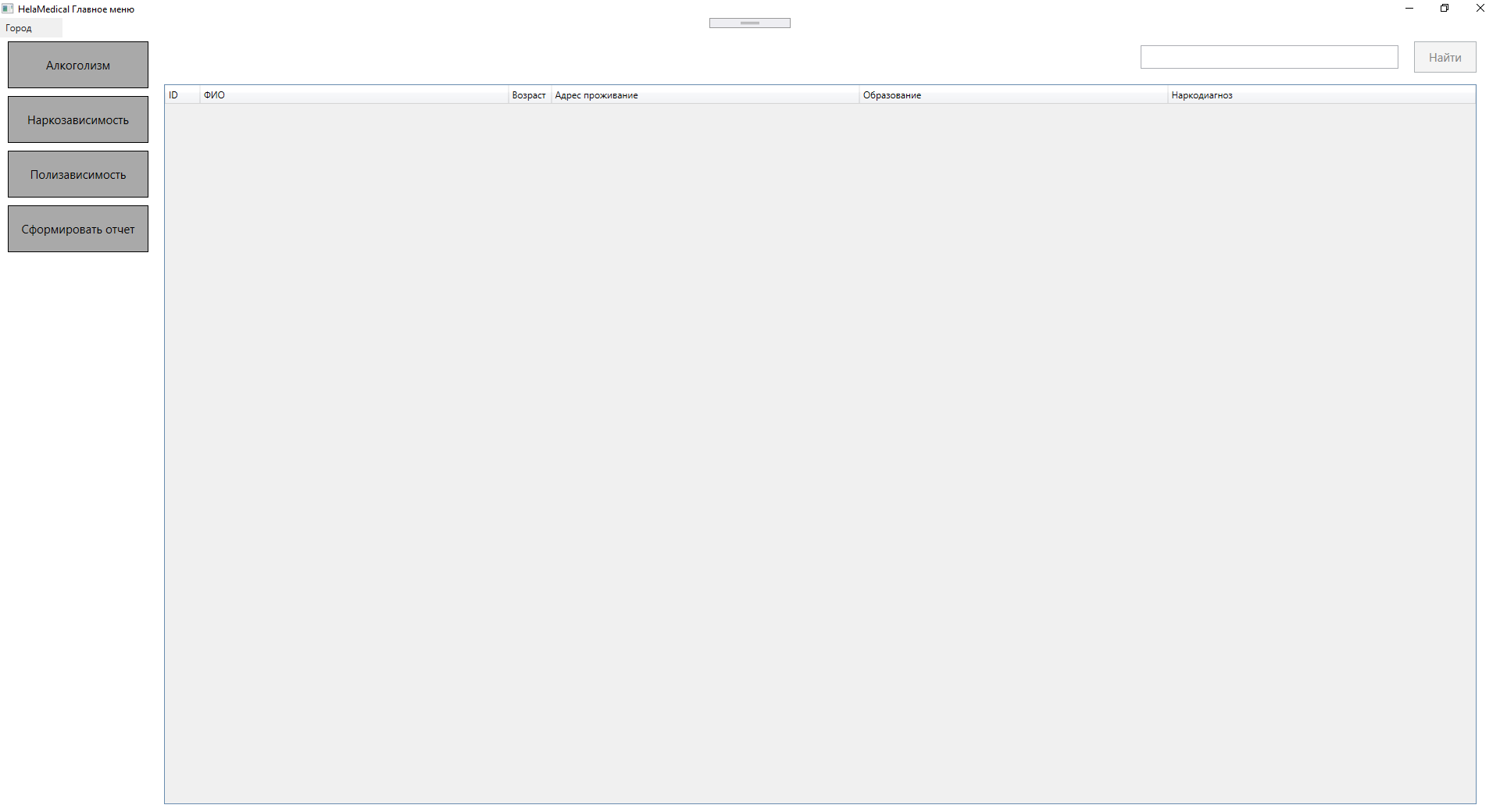


Рисунок 4.7 Окно создание персонала при не введенном поле персонал

После входа в систему отображается окно главного входа. Это окно разделено на 4 области. В первой области расположена ссылка на города. Ниже нее расположены клавиши на создание карт пациентов и формирование отчета.

В крайнем правом углу находиться TextBox, и клавиша найти. Клавиша найти не активна пока не ведут искомые данные в TextBox. Ниже клавиши найти находится DataGrid, в которой выводится информация по пациентам (смотрите рисунок 4.8)

Рисунок 4.8 Главное окно

При вводе фамилии в поле для поиска пациента и нажатии клавиши найти, программа производит поиск пациента в БД. При удачном поиске пациента в БД, программа выводит пациента в DataGrid, смотрите рисунок 4.9.

При удачной попытки поиска можно изменить некоторые критерия.

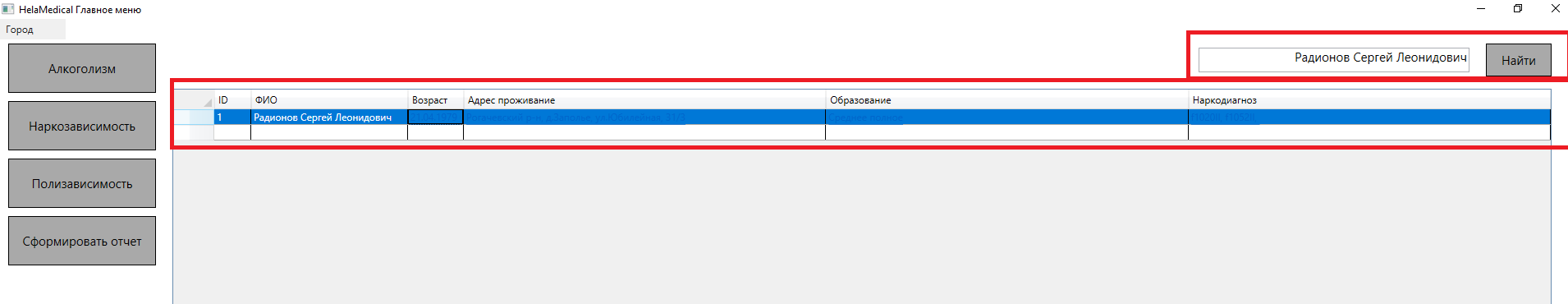


Рисунок 4.9 Главное окно. Вывод пациента в DataGrid

Если пациент на найден, то выведется соответствующее окно с ошибкой смотрите рисунок 4.10.

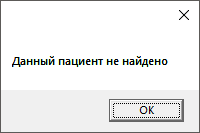


Рисунок 4.10 Окно ошибки при поиске пациента

При нажатии клавиши Город, открывается окно с областью и регионами. Где выбирается область и регион проживания пациента на которого заводится карточка. Смотрите рисунок 4.11.

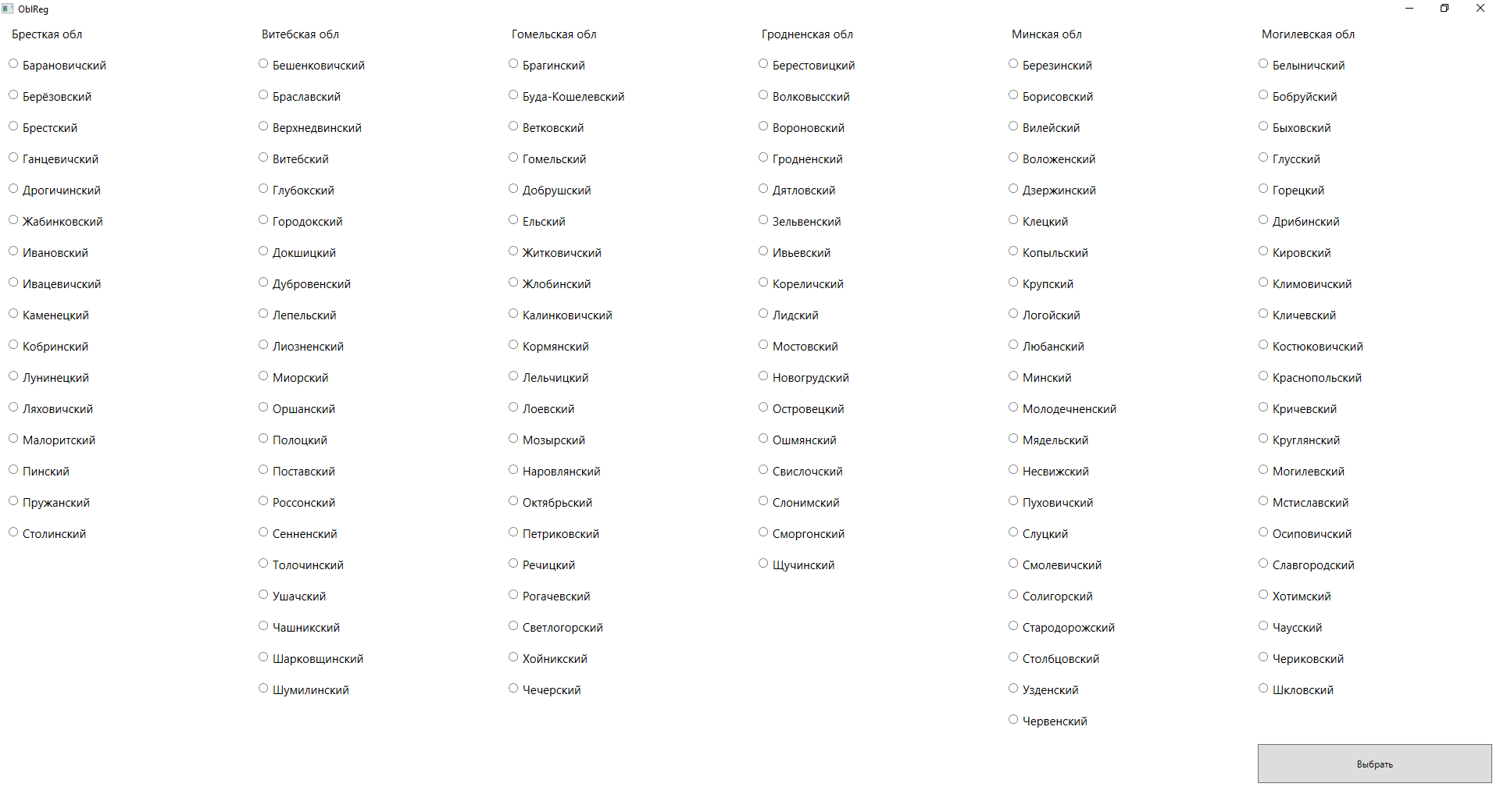


Рисунок 4.11 Окно выбора области и района.

После того как пользователь выбрал нужный район и нажал клавишу выдрать. Нужный район зафиксировался в программе до того, момента пока пользователю не потребуется его сменить.

При клике на клавишу Алкоголизм, наркозависимость или полизависимость, перед открытием новой формы для заполнения производится проверка, выбран ли область и регион, если район не был выбран то, программа откроет окно выбора области и района, только затем откроет окно формы заполнения карточки пациента. Если район выбран, то открывается новое окно с формой для заполнения индивидуальной карты пациента (см. рисунки 4.12-4.14).

Алгоритм работы функции поиска пациента реализована в приложение Б

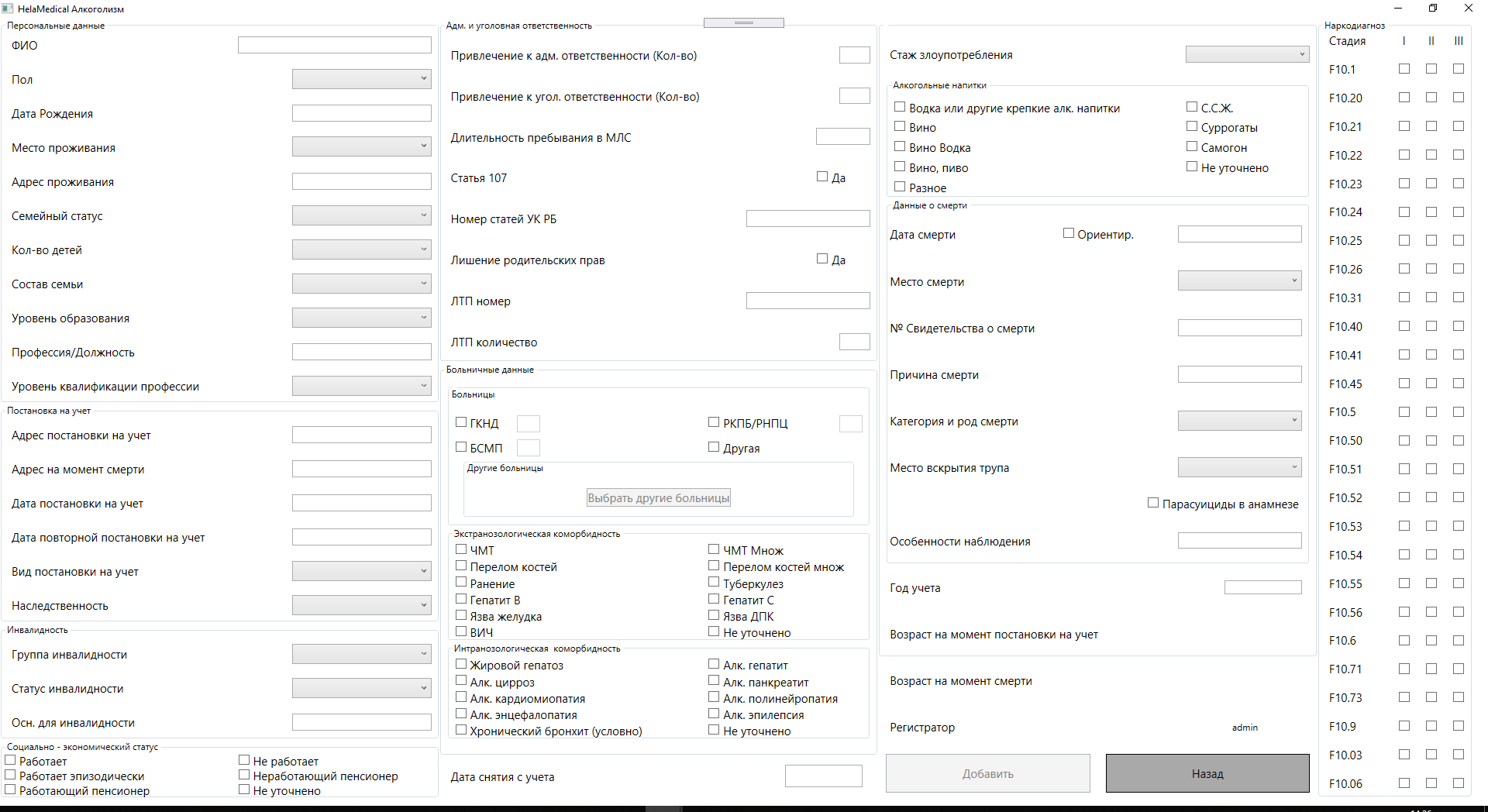


Рисунок 4.12 Окно ввода информации по пациенту

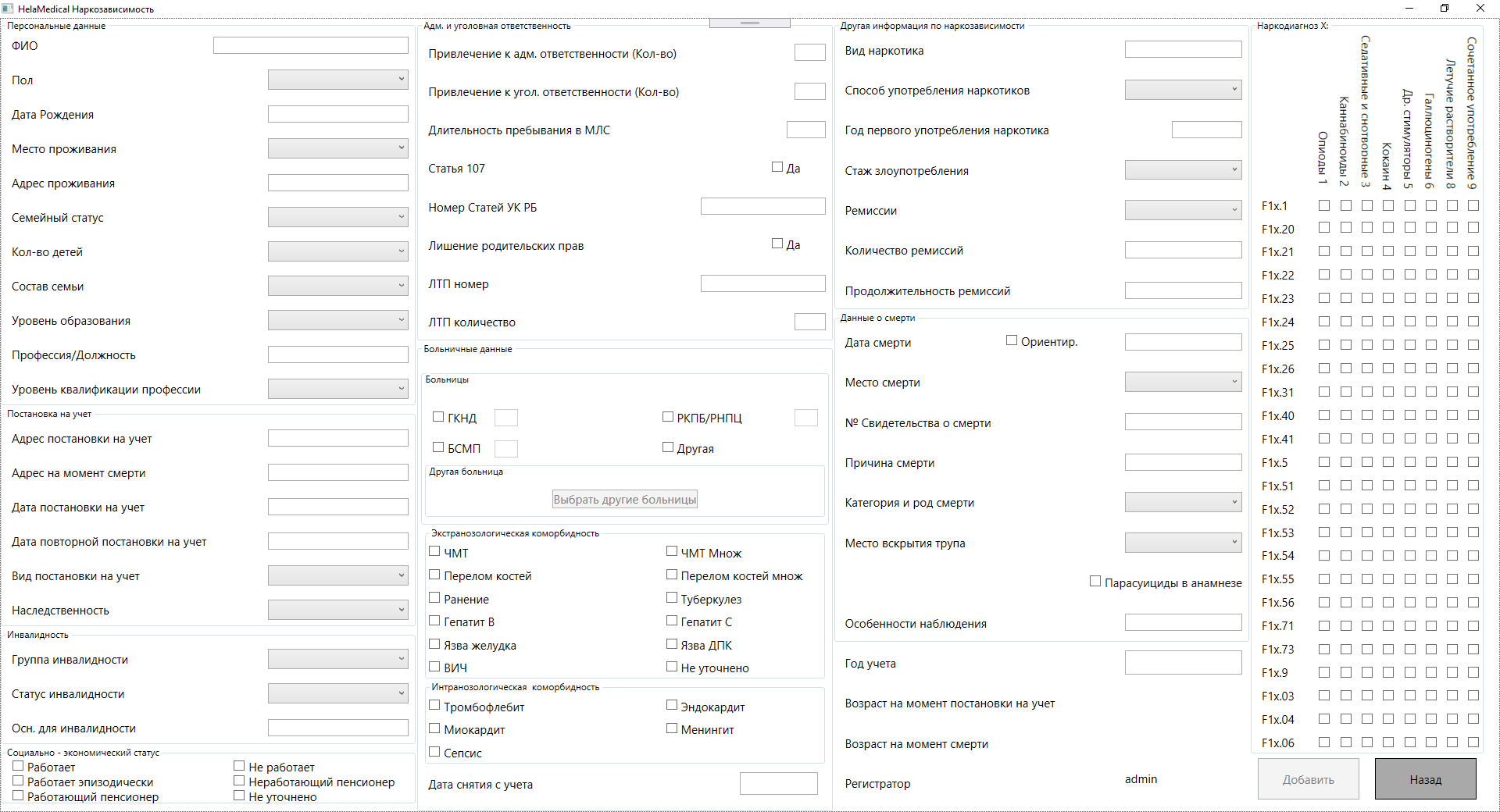


Рисунок 4.13 Окно ввода информации по пациенту

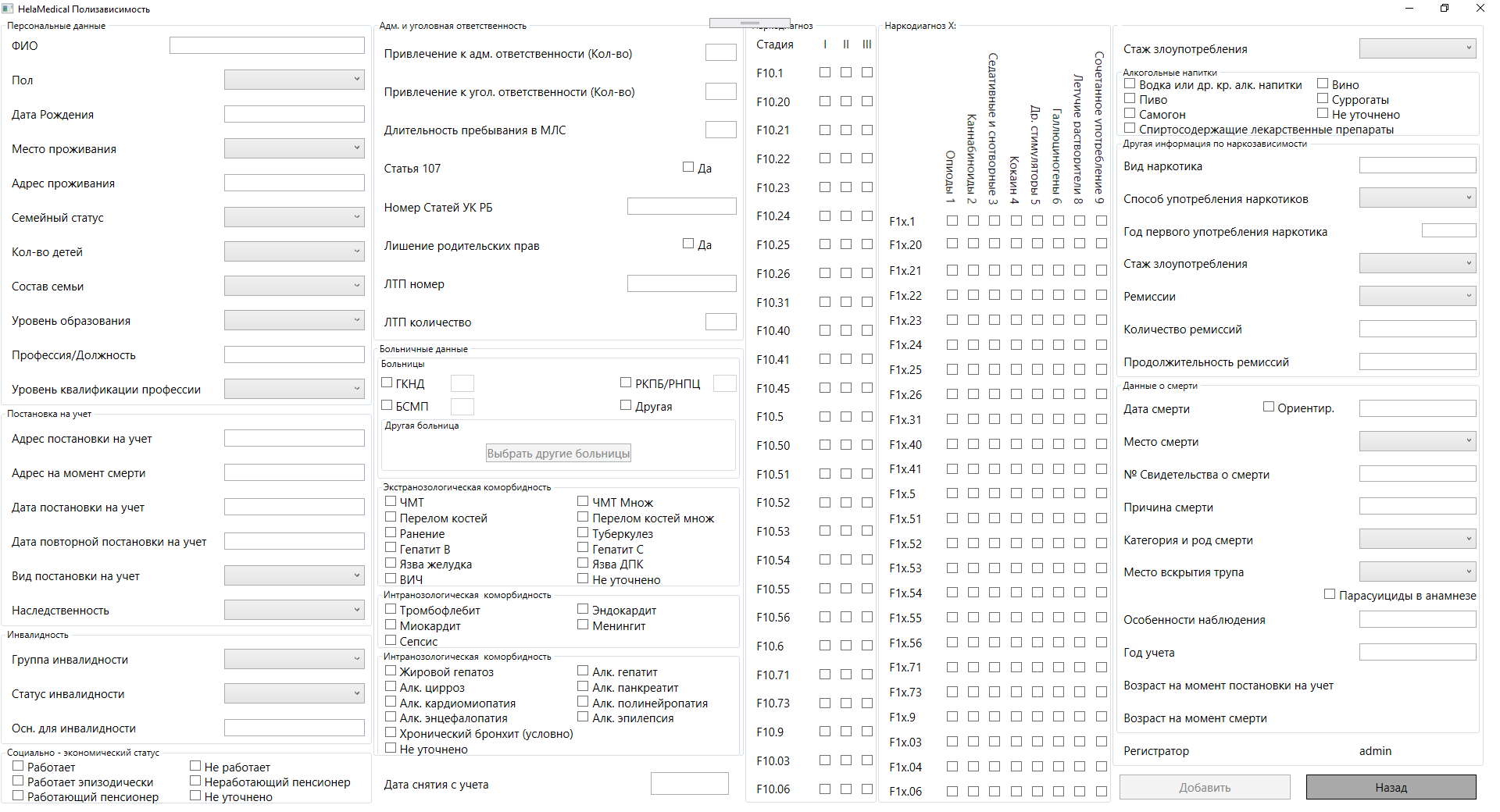


Рисунок 4.14 Окно ввода информации по пациенту

Если при создании карточки пациента, не больницы не соответствуют или у пациента намного больше предложенных больниц, то тогда ставиться на поле др. больницы галочка. Затем активируется клавиша выбора других больниц. Пользователю открывается окно для выбора других больниц смотрите рисунок 4.15.

При нажатии клавиши сформировать отчет открывается окно для формирования отчета (см. рисунок 4.16). Алгоритм формирования отчета

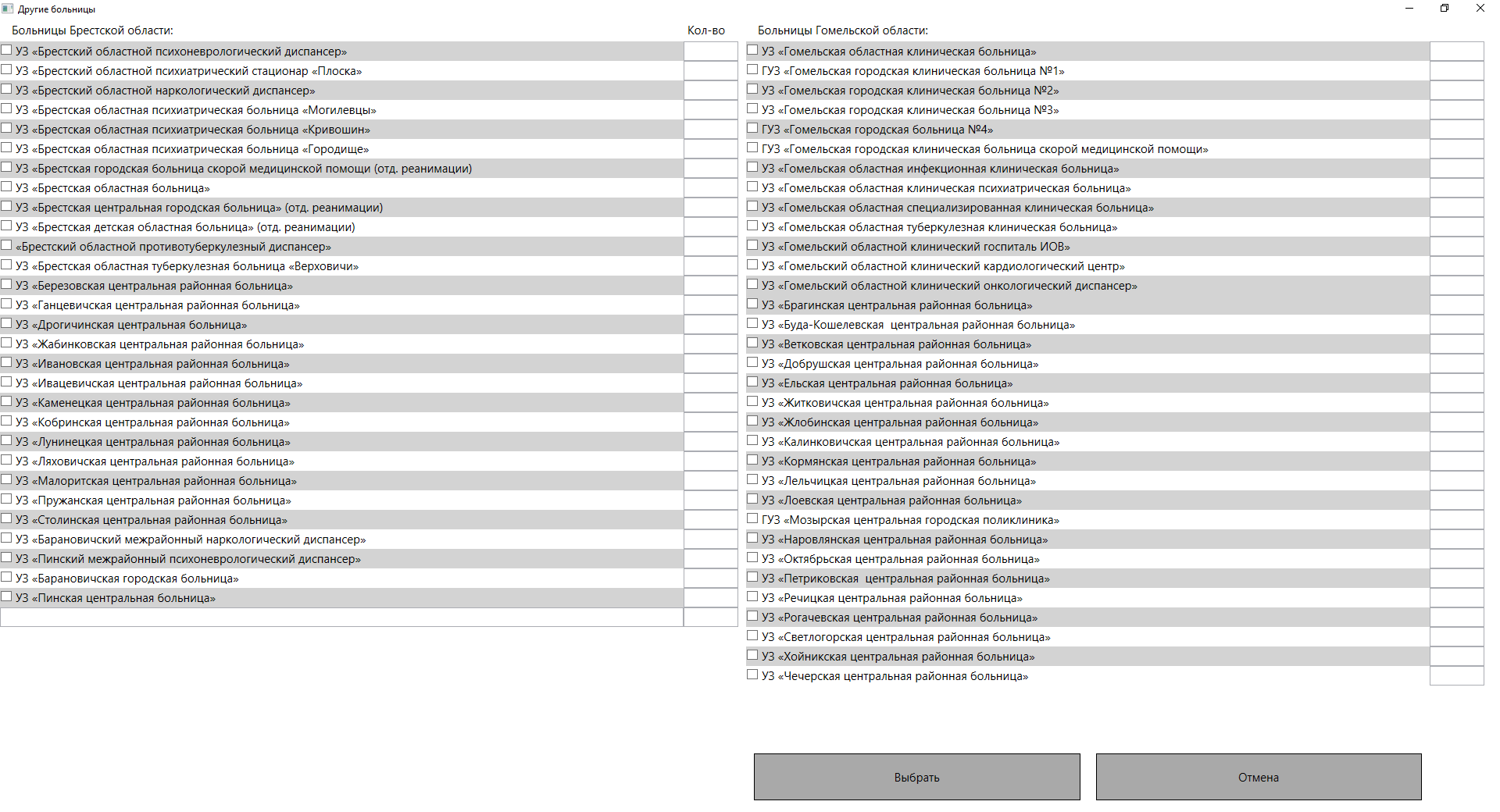


Рисунок 4.15 Окно выбора других больниц

представлен в приложении В. После выбора критериев формируется отчет согласно критериям. И выводится в DataGrid, для просмотра рисунок 4.17. Если не найдено пациентов по тем критериям поиска, то выводится окно с сообщением «Пациентов по данным критериям не найдено». Затем пользователь может всю информацию, которая появилась в DataGrid

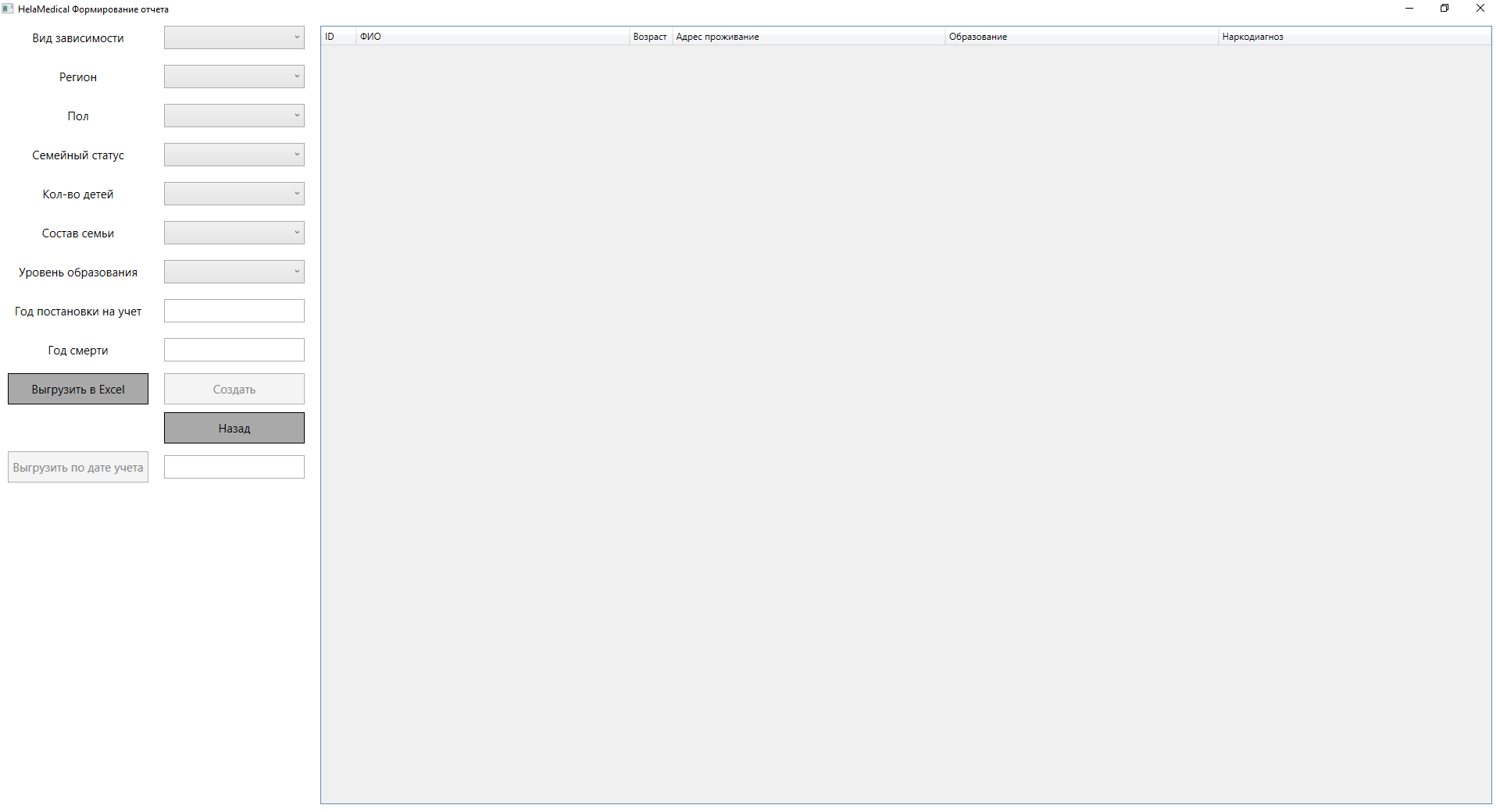


Рисунок 4.16 Окно формирования отчета и выгрузки информации в Excel

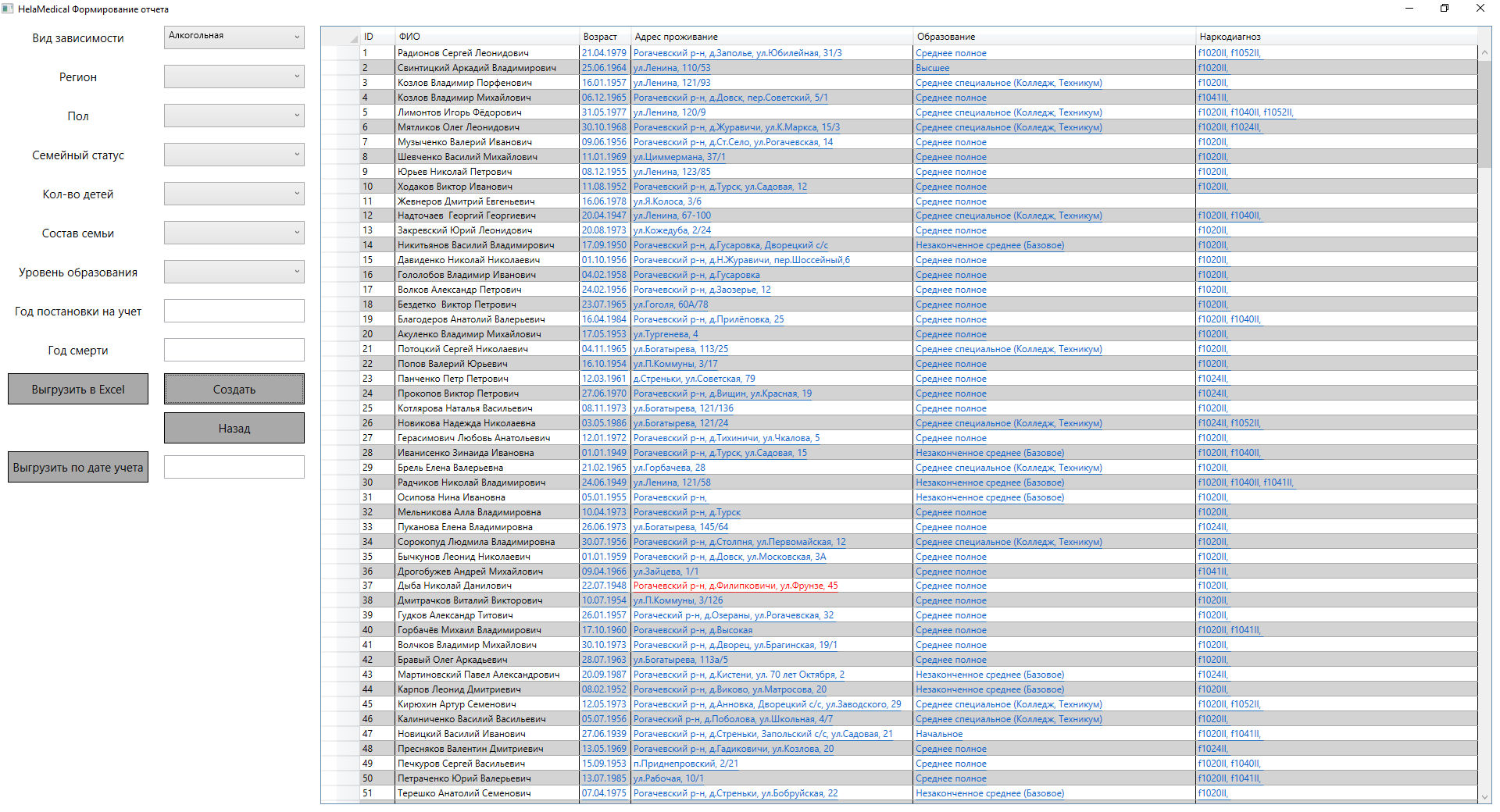


Рисунок 4.17 Поиск в БД по одному критерию

может выгрузить в Excel файл. Для этого ему необходима нажать на клавишу выгрузить в Excel. Программа произведет запись информации в Excel файл, если ни одного из критериев не будет выбрана, то клавиша создать не активна смотрите рисунок 4.18.

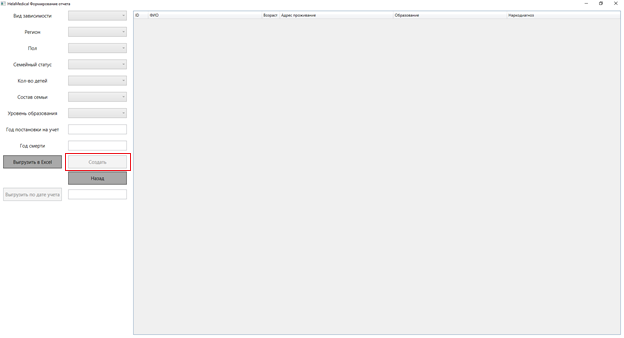
Для формирования отчета «учета по годам» необходимо выбрать вид зависимости пациента, затем выбрать год учета, затем нажать на клавишу выгрузить по дате учета. Затем программа просмотрит БД согласно 

Рисунок 4.18 Окно формирования отчета. Клавиша создать не активна

выборному виду зависимости пациента, затем отсортирует по году, который ввели поле TextBox. Затем программа формирует список пациентов с этим годом и отравляет его в метод, который записывает этот список в Excel файл и выводит соответствующее сообщение. Механизм представлен в приложение Г.

5 Тестирование разработанного программного обеспечения

**Тестирование программного обеспечения** — проверка соответствия между реальным и ожидаемым поведением программы, осуществляемая на конечном наборе тестов, выбранном определенным образом. В более широком смысле, тестирование — это одна из техник контроля качества.

**Качество программного обеспечения (Software Quality)** — это совокупность характеристик программного обеспечения, относящихся к его способности удовлетворять установленные и предполагаемые потребности.

Каскадная модель (Waterfall Model) является одной из наиболее старых моделей, которую можно применять не только для разработки или тестирования ПО (рисунок 5.1), но также практически для любого другого проекта. Его базовым принципом является последовательный порядок выполнения задач. Это значит, что мы можем переходить к следующему шагу разработки или тестирования только после того, как предыдущий был успешно завершен. Эта модель подходит для небольших проектов и применима только в том случае, если все требования точно определены. Главными достоинствами этой методологии являются экономическая эффективность, простота использования и управления документацией.

Процесс тестирования ПО начинается после завершения процесса разработки. На этой стадии все необходимые тесты переносятся с юнитов на системное тестирование для того, чтобы контролировать работу компонентов как по отдельности, так и в комплексе.



Рисунок 5.1 Каскадное тестирование

Помимо упомянутых выше достоинств, данный подход к тестированию также имеет и свои недостатки. Всегда существует вероятность обнаружения критических ошибок в процессе тестирования. Это может привести к необходимости полностью изменить один из компонентов системы или даже всю логику проекта. Но подобная задача невозможна в случае каскадной модели, поскольку возвращение на предыдущий шаг в этой методологии запрещено.

Для первой поставки программного обеспечения рекомендуется проводить Smoke + NFTAT готовой функциональности: поверхностное тестирование (Smoke Test) выполняется для определения пригодности сборки для дальнейшего тестирования; полное тестирование системы или ее части как на корректных, так и на некорректных данных/сценариях (Acceptance Test, AT) позволяет обнаружить дефекты.

Для последующих поставок программного обеспечения композиции тестов могут быть следующими.

Если не была добавлена новая функциональность, то DV + RTMAT. То есть, выполняется проверка исправления дефектов программистом (Defect Validation, DV), а также проверка работоспособности остальной функциональности после исправления дефектов на позитивных сценариях (Minimal Acceptance Test, MAT).

Если была добавлена новая функциональность, то Smoke + DV + NFTAT + RTMAT. В частности, выполняется поверхностное тестирование (Smoke Test), проверка исправления дефектов программистом (Defect Vali-dation, DV), тестирование новых функциональностей (New Feature Testing, NFT), проверка старых функциональностей, т. е. регрессионное тестирование (Regression Test).

Таблица 5.1 представлена вариант check-list для формирования программных тестов.

Если была добавлена новая функциональность, то возможен также вариант DV + NFTAT + RTMAT, т. е. без выполнения Smoke Test.

Таким образом, для второй и последующих поставок обобщенная схема композиции тестов выглядит следующим образом:

Smoke + DV + (NFTAT) + RTMAT.

Таблица 5.1 Check-list

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Module/submodule** | **Actions** | **Test type** |
|  | **ПРИЛОЖЕНИЕ** |  |  |
| **1.** |  | Запуск приложения | Smoke |
| **2.** |  | Авторизация пользователя в приложении | Smoke |
|  | **Главное окно** |  |  |
| **3.** |  | GUI | Smoke |
| **4.** | **Главное окно** | ФТ-1: Выбор области и района | Smoke |
| **5.** | ФТ-1: Выбор типа зависимости пациента | Smoke |
| **6.** | ФТ-1: Поиск пациента в БД по ФИО | AT |
|  |  |  |
| **7** | **Ввод данных пациента в карточку:** |  |  |
| **8** | Ввод персональных данных пациента | MAT |
| **9** | Выбор чекбокса другие, для активации клавиши другие больницы | AT |
| **10** | Открытие окна другие больницы | МАТ |
| **11** | Ввод в поле год рождение не числового значение | MAT |
| **12** | Ввод в дату регистрации не числового значения | МАТ |
| **13** | Ввод в дату смерти не числового значения | АТ |
| **14** | ввод в кол-во больниц не числового значения | АТ |
| **15.** | ввод в количество больниц < 0 | AT |
| **16.** | ввод в год учета на числового значения | АТ |
| **17.** | не выбрать область и район | АТ |
| **18.** | ввод любого нечислового значения в поле "Количество" | АТ |
| **19.** | **Сформировать отчет** | не выбрать тип зависимости | МАТ |
| **20.** | не выбрать регион | АТ |
| **21.** | не выбрать уровень образования | АТ |
| **22.** | не ввести дату в год регистрации | МАТ |
| **23.** | формирование отчета в Excel файле | МАТ |

В зависимости от типа и специфики приложения (web, desktop, mobile) выполняют специализированные тесты (например, кроссплатформенное тестирование). В таблице 5.2 представлен Test- list на данную программу

Таблица 5.2 Test-list

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Module** | **Test Case Headline + Description** | **Expected Results** |
| **1** | Приложение | **Запуск приложения** 1. Запустить приложение | 1. Отображается главная страница приложения Ввод логина и пароля, регистрация нового пользователя |
| **2** | Приложение | **Авторизация пользователя в приложении** 1. Авторизоваться в приложении | 1 Открытие главного окна |
| **3** | Окно регистрации | Регистрации пользователя в приложении | 1 Ввод ФИО 2 Ввод должности 3 Ввод пароля  4 Ввод повторного пароля |
| **4** | Главное окно | Открыто главное окно выбора создания пациента, выбор района *Предусловия: Пользователь залогинен в приложении*  1. Выбрать район и область пациента  2. Выбрать вид пациента 3. Ввод ФИО в окне поиска пациента 4. нажатие клавиши найти | 1. Отображается в верхнем левом углу клавиша город 2. Слева три клавиши выбора категории пациента 3. Снизу клавиша формирования отчета  4 с правом верхнем углу находится клавиша поиска пациента по ФИО 5. В центре в дата гриде выводить информация о пациенте |
| **5** | Окно Алкоголики | **Заполнение карточки пациента** *Предусловия: Пользователь залогинен в приложении пользователь выбрал создание карты пациентов алкоголики* 1. Заполнение полей карты пациента 2. Заполнение поля ФИО  3. Заполнение личной информации пациента.  4. По необходимости активировать чекбокс в поле другие больницы. 5. Выбор других больниц в новом окне после нажатия клавиши другие больницы 6. нажатия клавиши добавить 7. Вывод MessageBox о том что произошла запись информации в БД | 1. Поле ФИО заполнено. Активируется клавиша добавить  2. заполнение личной информации по пациенту  систем пропускать создания карточки. 3. При активации чекбокса другие в поле больницы, активируется клавиша другие больницы,. 4. При нажатии клавиши другие больницы открываются окно с информации о других больницах 5. После нажатия клавиши добавить. Информация записывается в БД. 6. Вывод сообщения о записи в БД |

Продолжение таблицы 5.2

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Module** | **Test Case Headline + Description** | **Expected Results** |
| **6** | Окно Наркоманы | **Заполнение карточки пациента** *Предусловия: Пользователь залогинен в приложении пользователь выбрал создание карты пациентов наркоманы* 1. Заполнение полей карты пациента 2. Заполнение поля ФИО  3. Заполнение личной информации пациента.  4. По необходимости активировать чекбокс в поле другие больницы. 5. Выбор других больниц в новом окне после нажатия клавиши другие больницы 6. нажатия клавиши добавить 7. Вывод MessageBox о том что произошла запись информации в БД | 1. Поле ФИО заполнено. Активируется клавиша добавить  2. заполнение личной информации по пациенту  систем пропускать создания карточки. 3. При активации чекбокса другие в поле больницы, активируется клавиша другие больницы,. 4. При нажатии клавиши другие больницы открываются окно с информации о других больницах 5. После нажатия клавиши добавить. Информация записывается в БД. 6. Вывод сообщения о записи в БД |
| **7** | Окно Полизависымые | **Заполнение карточки пациента** *Предусловия: Пользователь залогинен в приложении пользователь выбрал создание карты пациентов полизависимые* 1. Заполнение полей карты пациента 2. Заполнение поля ФИО  3. Заполнение личной информации пациента.  4. По необходимости активировать чекбокс в поле другие больницы. 5. Выбор других больниц в новом окне после нажатия клавиши другие больницы 6. нажатия клавиши добавить 7. Вывод MessageBox о том что произошла запись информации в БД | 1. Поле ФИО заполнено. Активируется клавиша добавить  2. заполнение личной информации по пациенту  систем пропускать создания карточки. 3. При активации чекбокса другие в поле больницы, активируется клавиша другие больницы,. 4. При нажатии клавиши другие больницы открываются окно с информации о других больницах 5. После нажатия клавиши добавить. Информация записывается в БД. 6. Вывод сообщения о записи в БД |

Продолжение таблицы 5.2

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Module** | **Test Case Headline + Description** | **Expected Results** |
| **8** | Окно сформировать отчет | Сформировать отчет  *Предусловия: Пользователь залогинен в приложении пользователь выбрал создание карты пациентов полизависимые* 1. выбор типа зависимости пациента в комбобоксе 2. Нажать клавишу создать 3. Повторить пункт 1, затем выбор региона проживание пациента в комбобоксе повторить пункт 2. 4. Повторить пункт 1, затем выбирать семейного статуса пациента в комбобоксе Затем повторить пункт 2.  5. Повторить пункт 1, затем выбирать количество детей пациента в комбобоксе повторить пункт 2. 6. Повторить пункт 1, затем выбрать состав семьи в комбобоксе повторить пункт 2 7. Повторить пункт 1, затем выбрать уровень образования в комбобоксе повторить пункт 2 8. Повторить пункты 1 введите в поле год постановки на учет повторить пункт 2 9. повторить пункт 1, затем введите в поле год смерти, повторите пункт 2 10. нажать клавишу выгрузить в Excel | 1. Выбор вид зависимости  2. Вывод информация в датагрид. 3. выбор региона проживание пациента в комбобоксе Вывод информации в датагрид. 4. выбирать семейного статуса пациента в комбобокс. Вывод информации в датагрид  5. выбирать количество детей пациента в комбобоксе вывод информации в датагрид. 6. выбрать состав семьи в комбобоксе Вывод информации в датагрид 7. Выбрать уровень образования в комбобоксе Вывод информации в датагрид 8. Введите в поле год постановки на учет Вывод информации в датагрид 9. Введите в поле год смерти, Вывод информации в датагрид 10. При нажатии клавиши выгрузить в Excel, вся информация переноситься из датагрид в Excel файл |

Заключение

Разработанная программная система направлена на реализацию потребностей специалистов по учету пациентов с наркологической зависимостью, в программном продукте охватываются области учета и формирования отчетов по различным видам зависимостям, различным областям и некоторым функциям. Это снизит временные затраты на обработку и внесение информации, а также ускорит процесс получения в виде необходимых отчетов.

Определены основные инструменты разработки и требования к программной системы. Применяемая система WPF позволила максимально абстрагировать интерфейс, предоставляемый пользователю от логики приложения, что в перспективе упростит интеграцию программной системы в уже существующие на предприятии и повысит востребованность программного продукта. Технология «Code First» позволила генерировать таблицы и их отношения на основе взаимоотношения классов и установленных атрибутов, что в значительной степени снизило время разработки базы данных.

Выбранная многослойная архитектура за счет низкой связанности слоев приложения и простоты управления положительно повлияло на абстрагирование представления программного продукта от логики. Каждый слой приложения имеет набор классов для реализации ее функций. Слои взаимодействуют при помощи, специальных сервисов, которые содержат методы для взаимодействия с нижележащими слоями.

Разработанные типовые сценарии выполнения основных предикатов на основе диаграмм activity ролей пользователей охватывают все требования к программной системе в части ее функциональности. Сценарий реализации методов взаимодействия с базой данных и работе с данными из базы данных (формирование отчетов и типов представления) являются основными направлениями программной системы.

Перспективными планами по направлению функциональности системы является формирования обще республиканской базой данных по нарко завимых людей. Формирование отчетов для общей республиканской статистики.

Список использованных источников

1. Абрамян Михаил Visual C# на примерах; БХВ-Петербург - М., 2008. - **317** c.

2. Албахари Джозеф , Албахари Бен C# 6.0. Справочник. Полное описание языка; Вильямс - М., 2016. - **623** c.

3. Биллиг В. А. Основы программирования на С#; Интернет-университет информационных технологий, Бином. Лаборатория знаний - М., 2012. - 488 c.

4. Васильев Алексей C#. Объектно-ориентированное программирование; Питер - М., 2012. - 320 c.

5. Дейтел П. Как программировать на Visual C# 2012; Питер - М., 2014. - **316** c.

6. Джозеф Албахари C# 5.0. Справочник. Полное описание языка; Диалектика / Вильямс - М., 2014. - **545** c.

7. Евдокимов П. С# на примерах; **ИЛ**- Москва, 2016. - **865** c.

8. Культин Никита Основы программирования в Microsoft Visual C# 2010; БХВ-Петербург - М., 2011. - **634** c.

9. Лотка Р. C# и CSLA .NET Framework. Разработка бизнес-объектов; Диалектика / Вильямс - М., 2010. - **485** c.

10. Мартин Р. С., Мартин М. Принципы, паттерны и методики гибкой разработки на языке C#; Символ-плюс - М., 2011. - 768 c.

11. Мэтью Мак-Дональд WPF: Windows Presentation Foundation в .NET 4.5 с примерами на C# 5.0 для профессионалов; Диалектика / Вильямс - М., 2013. - **735** c.

12. Аллен, Г. Тейлор SQL для чайников / Аллен Г. Тейлор. - М.: Диалектика, Вильямс, **2015**. - 416 c.

13. Бен, Форта SQL за 10 минут / Форта Бен. - М.: Диалектика / Вильямс, 2015. - 0 c.

14. Грабер, Мартин SQL для простых смертных / Мартин Грабер. - М.: ЛОРИ, 2014. - 378 c.

15. Гудсон, Джон Практическое руководство по доступу к данным (+ DVD-ROM) / Джон Гудсон , Роб Стюард. - М.: БХВ-Петербург, 2013. - 304 c.

16. Дейт, К. Дж. SQL и реляционная теория. Как грамотно писать код на SQL / К.Дж. Дейт. - М.: Символ-плюс, **2016**. - 0 c.

17. Джеймс, Р. Грофф SQL. Полное руководство / Джеймс Р. Грофф, Пол Н. Вайнберг, Эндрю Дж. Оппель. - М.: Вильямс, 2014. - 960 c.