|  |  |  |
| --- | --- | --- |
| |  |  | | --- | --- | | Логотип КБК копия | **Автономная некоммерческая организация профессионального образования**  **КАЛИНИНГРАДСКИЙ БИЗНЕС-КОЛЛЕДЖ** |   Кафедра Информационные системы и программирование |

Кафедра Информационных технологий

**Курсовая работа**

На тему: **«База данных медицинского кооператива»**

# По дисциплине: МДК 01.01 Разработка программных модулей

Выполнил студент

группы 23-ИСП-3/1

Идрисов Э.М.

Проверил:  
Зав. Каф. Дубинин А. В.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Калининград**

**2024**

**Содержание**

**Оглавление**

[По дисциплине: МДК 01.01 Разработка программных модулей 1](#_Toc184857212)

[Введение 3](#_Toc184857213)

[Задание 3](#_Toc184857214)

[1. Разработка Базы Данных 4](#_Toc184857215)

[2. Создание пользовательского интерфейса 6](#_Toc184857216)

[3. Основные задачи 15](#_Toc184857217)

[4. Заключение 23](#_Toc184857218)

# Введение

В современной медицинской сфере, как и во многих других отраслях, автоматизация играет ключевую роль. Об этом свидетельствует пример медицинского кооператива. Кооперативу понадобилась собственная база данных и удобный интерфейс для работы с ней.

Целью курсовой работы является создание собственной базы данных для медицинского кооператива «StalMed» под его конкретные задачи и написание WPF-приложения с использованием языка C# и Entity Framework с помощью паттерна проектирования MVVM.

# Задание

Медицинский кооператив «StalMed» предоставляет услуги по оказанию амбулаторной и консультативной медицинской помощи. Пациенты, обращающиеся в кооператив, записываются на приёмы к врачам различных специальностей. Врачи, работающие в кооперативе, являются медицинскими специалистами, оказывающими услуги в рамках своих компетенций.

Один и тот же врач может вести приёмы в разные дни и время, а один и тот же пациент может посещать приёмы у разных врачей или повторно у одного и того же.

В базе данных кооператива фиксируется информация о каждом приёме: дата проведения, врач, пациент, диагноз, симптомы и назначенное лечение. Также ведётся учёт врачей (ФИО, логин, пароль) и пациентов (ФИО, дата рождения, пол).

Пациент может записываться на любое количество приёмов, а врач — принимать любое количество пациентов. Один и тот же человек может быть зарегистрирован как пациент, а в другом контексте — как сотрудник кооператива, однако в рамках учётной системы роли строго разделены.

После каждого приёма в систему вносятся данные о поставленном диагнозе, назначенном лечении и других результатах визита пациента.

# Разработка Базы Данных

Создание Базы Данных начнется с нормализацией бд, после третий нормальной формы можно приступить с построение. Определяются нужные таблицы с первичными и внешними ключами для связки таблицы, а также типы данных каждого поля.

* Patient — эта таблица отвечает за пациентов. В неё вносятся данные о каждом пациенте: имя (patient\_name), пол (pol), дата рождения (data\_rozdeniya), адрес (adres) и уникальный идентификатор (patient\_id)
* Doctor — эта таблица отвечает за врачей. Содержит поля: имя врача (doctor\_name), логин (doctor\_login), пароль (doctor\_password) и уникальный идентификатор (doctor\_id)
* Drug — эта таблица отвечает за лекарства. Включает поля: название препарата (drug\_name), способ приёма (sposob\_priema), предполагаемое действие (deistvie), побочные эффекты (pobochnie\_effects) и уникальный идентификатор (drugs\_id)
* Priem — эта таблица отвечает за приёмы пациентов. Содержит такие поля, как дата и время приёма (data), место проведения (mesto), симптомы (simptomi), диагноз (diagnoz), назначение (naznachenie), а также внешние ключи: patient\_id, drugs\_id, doctor\_id и priem\_id.

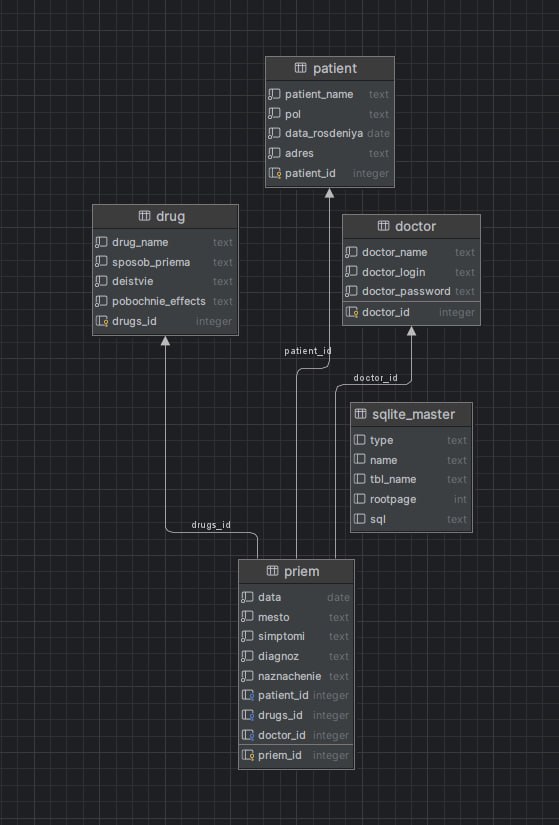


Рис.1 Схема базы данных для фирмы

# Создание Архитектуры приложения

# 2.1 Паттерн MVVM

Паттерн **MVVM (Model-View-ViewModel)** позволяет отделить логику приложения от визуальной части (представления). Данный паттерн является архитектурным, то есть он задает общую архитектуру приложения. Данный паттерн был представлен Джоном Госсманом (John Gossman) в 2005 году как модификация шаблона Presentation Model и был первоначально нацелен на разработку приложений в WPF. И хотя сейчас данный паттерн вышел за пределы WPF и применяется в самых различных технологиях, в том числе при разработке под Android, iOS, тем не менее WPF является довольно показательной технологией, которая раскрывает возможности данного паттерна.

MVVM состоит из трех компонентов: модели (Model), модели представления (ViewModel) и представления (View).



Рис. 2 Взаимодействие в MVVM

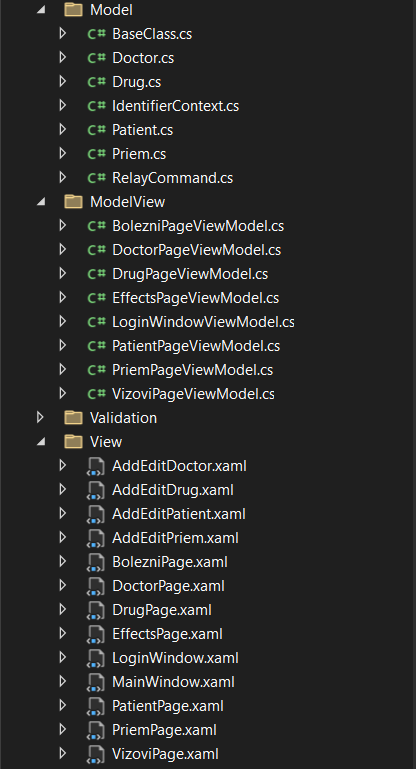
Модель (Model) описывает используемые в приложении данные. Модели могут содержать логику, непосредственно связанную этими данными, например, логику валидации свойств модели. В то же время модель не должна содержать никакой логики, связанной с отображением данных и взаимодействием с визуальными элементами управления.

Представление (View) определяет визуальный интерфейс, через который пользователь взаимодействует с приложением. Применительно к WPF представление - это код в xaml, который определяет интерфейс в виде кнопок, текстовых полей и прочих визуальных элементов.

Модель представления (ViewModel) связывает модель и представление через механизм привязки данных. Если в модели изменяются значения свойств, при реализации моделью интерфейса INotifyPropertyChanged автоматически идет изменение отображаемых данных в представлении, хотя напрямую модель и представление не связаны. ViewModel также содержит логику по получению данных из модели, которые потом передаются в представление. И также VewModel определяет логику по обновлению данных в модели.

## **2.2 Реализация MVVM**

**Архитектура продукта имеет следующий вид:**



# Создание пользовательского интерфейса

Первое окно, которое нас встречает это окно регистрации/входа. В нем вы сначала создаете свое пользователя (с помощью кнопки регистрация) он сохраняется в бд, а после вы нажимаете кнопку вход и вам открывается полное взаимодействие с приложением

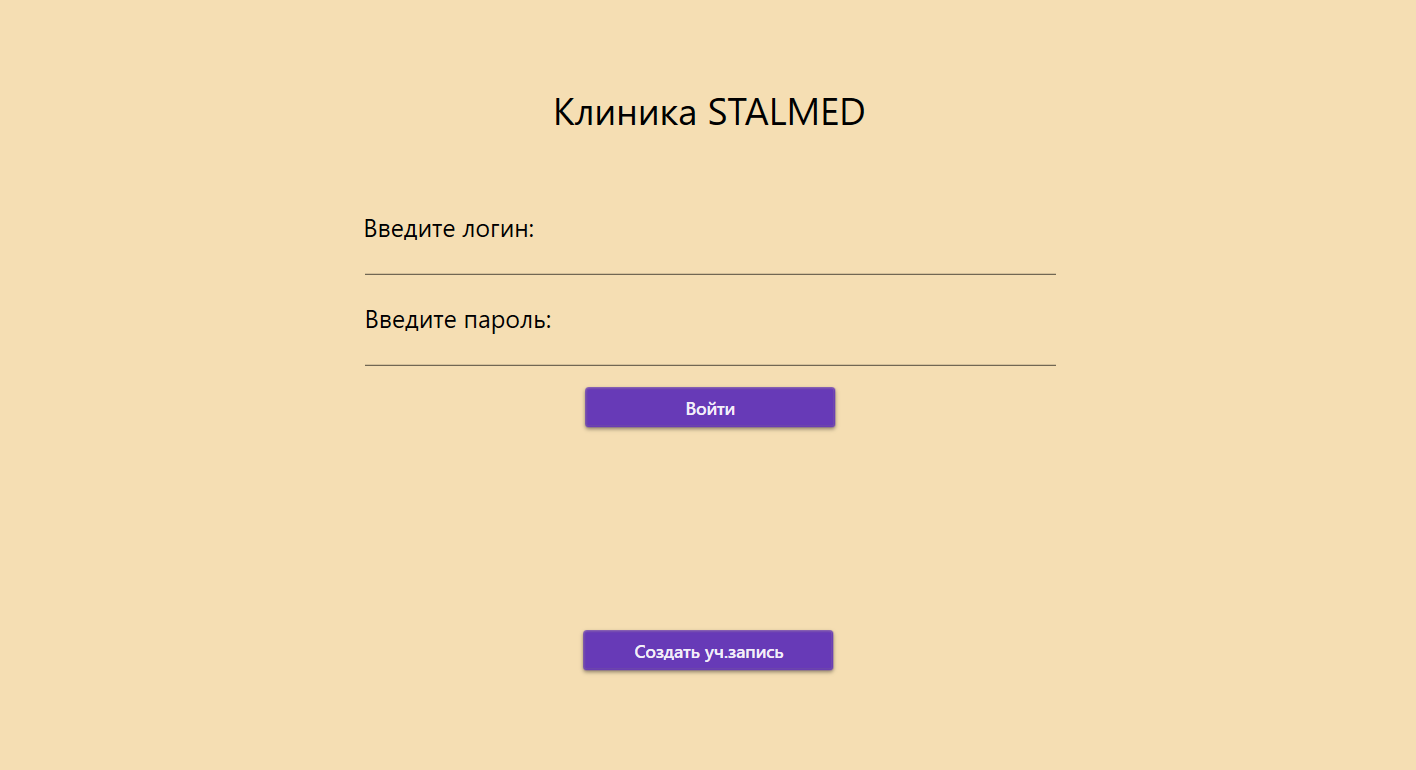


Рис.2 Начальное окно Регистрации/Входа

После Успешной регистрации/входа откроется Главное Меню.

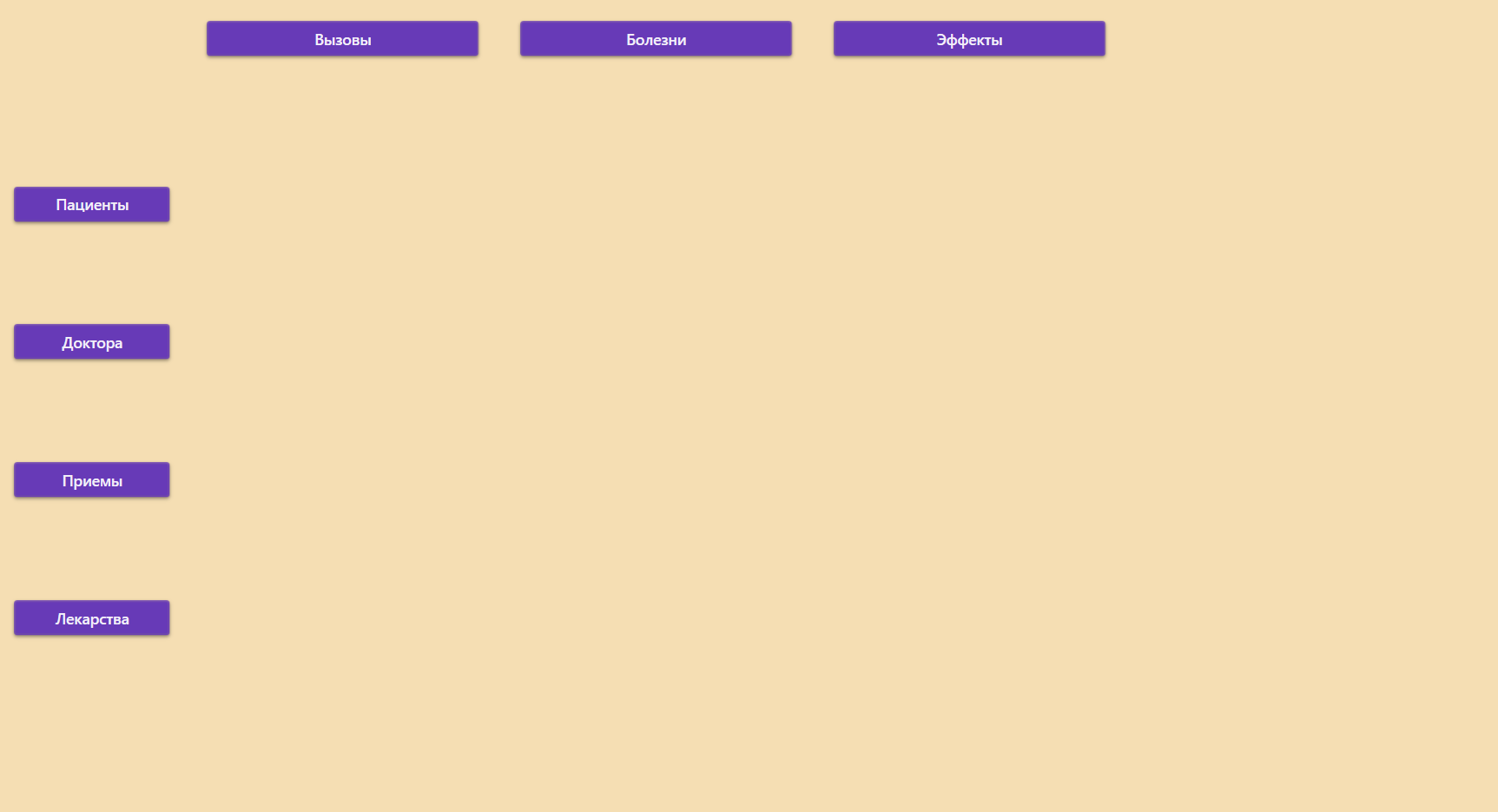


Рис.3 Главное Меню

В левой части окна находятся кнопки с именем таблиц базы данных. В верхней и нижней части помещены кнопки для запросов. В середине находится рабочая места, где происходит просмотр записи таблиц и взаимодействие с этими объектами в бд. Для каждой таблицы созданы отдельные окна, при нажатии на кнопку с именем таблиц вам доступно будет таблица, где реализованы инициализация и просмотр записей соответствующей таблицы.

В этом окне происходит инициализация всех Приемов и взаимодействие с этими объектами в бд (Create/Delete)

Здесь вы взаимодействуйте с аукционами по средствам кнопок и полей, также реализованы конверторы для более удобной работы.

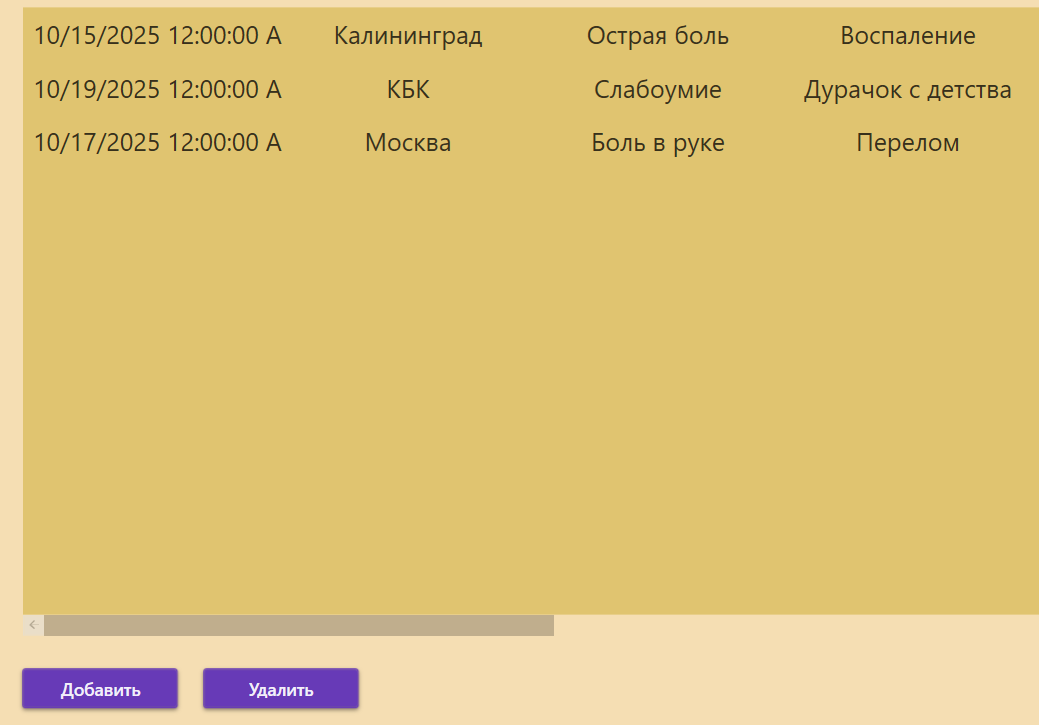


Рис. 3 Окно Приемов, отображение всех приемов

Для того чтобы удалить запись из таблицы нужно нажать на кнопку Удалить

Для того чтобы добавить запись в таблицу нужно нажать на кнопку Добавить и появится окно для добавления и изменения записи

Также стоит отметить, что редактирование записей реализовано двойным щелчком по элементу.

Это окно добавления сотрудников в этом окне происходит редактирование всех сотрудников и взаимодействие с этими объектами в БД (создание/удаление)

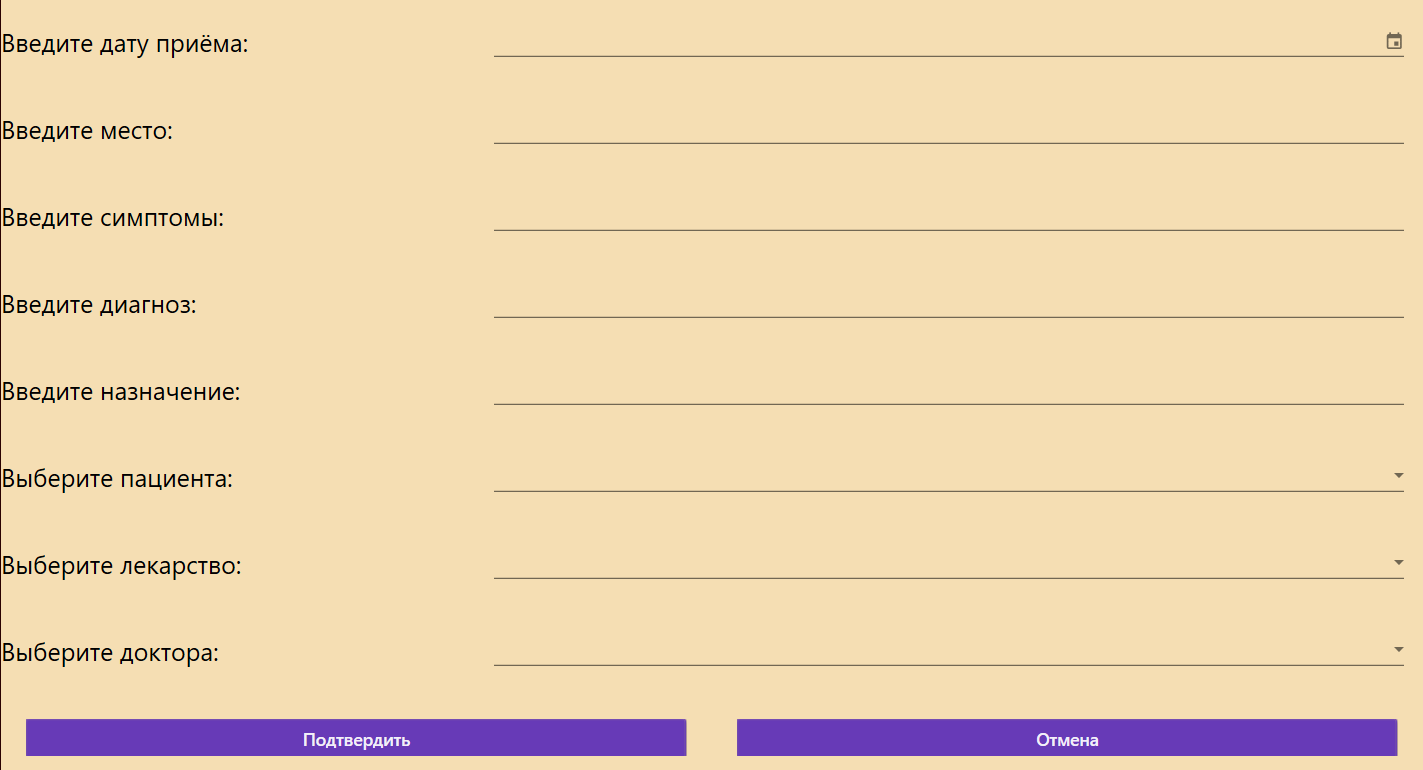


Рис. 4 Окно редактирование Приемов

Это окно добавления лотов в этом окне происходит редактирование всех приемов и взаимодействие с этими объектами в БД (создание /удаление/изменение). Здесь можно изменить дату, Место, симптомы, диагноз, назначенное лекарство, доктора и пациента . При нажатии кнопки сохранить запись сохраняется в БД. Если дважды нажать на поле из списка то можно будет изменить его и все изменения сохранятся в БД

В этом окне происходит инициализация всех Приемов и взаимодействие с этими объектами в бд (Create/Delete)

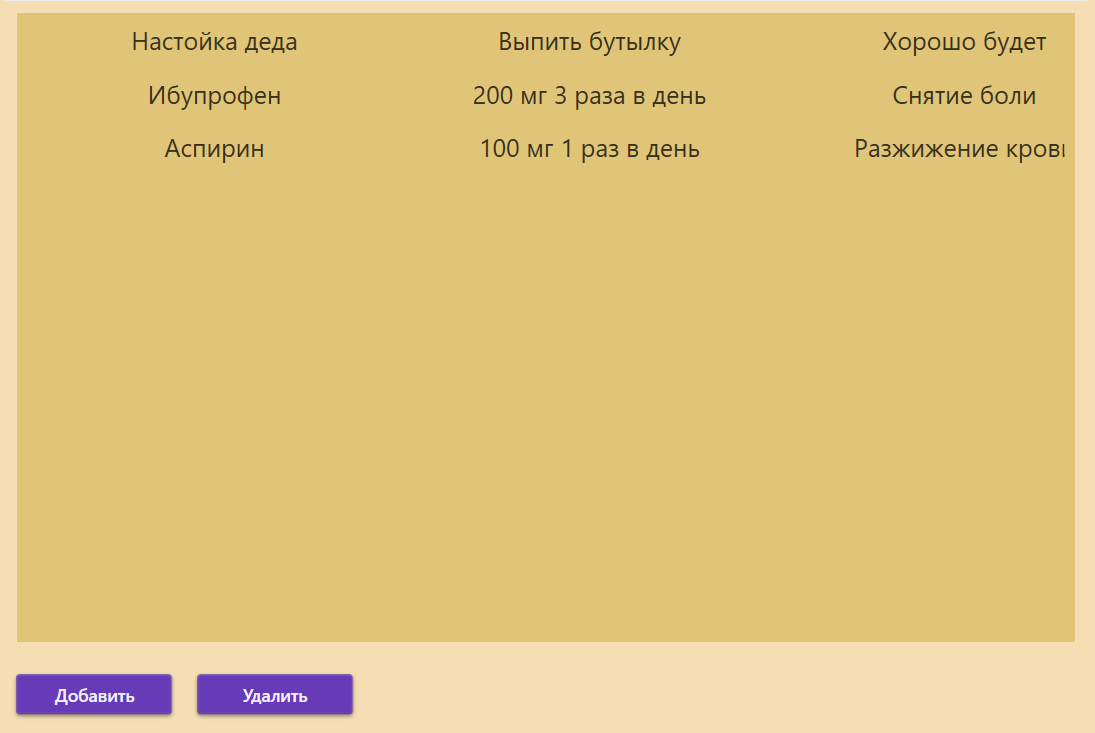
Здесь вы взаимодействуйте с лотами по средствам кнопок и полей 

Рис.5 Окно лекарств, отображение всех записей

Для того чтобы удалить запись из таблицы нужно нажать на кнопку Удалить

Для того чтобы добавить запись в таблицу нужно нажать на кнопку Добавить и появится окно для добавления и изменения записи

Также стоит отметить, что редактирование записей реализовано двойным щелчком по элементу.

Это окно добавления сотрудников в этом окне происходит редактирование всех сотрудников и взаимодействие с этими объектами в БД (создание/удаление)

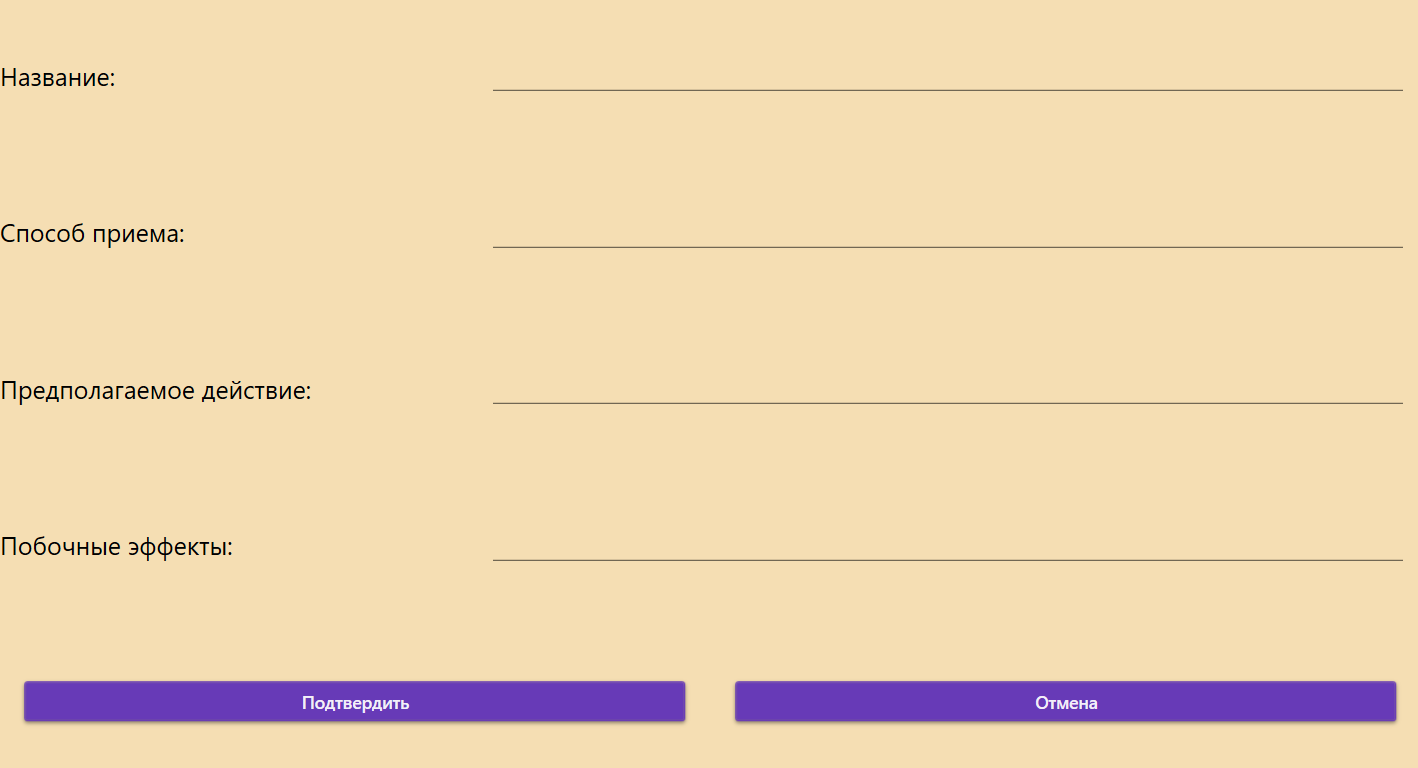


Рис.6 Редактирование записей таблицы лекарств

Это окно добавления/редактирования приёмов в этом окне происходит редактирование всех приёмов и взаимодействие с этими объектами в БД (создание / удаление / изменение). Здесь можно изменить дату и время приёма, врача, пациента, диагноз и описание состояния. При нажатии кнопки «Подтвердить» запись сохраняется в БД. Если дважды нажать на поле из списка, то можно будет изменить его — все изменения сохраняются в БД.

В этом окне происходит инициализация всех пациентов.

Здесь используются конверторы для более понятной информации

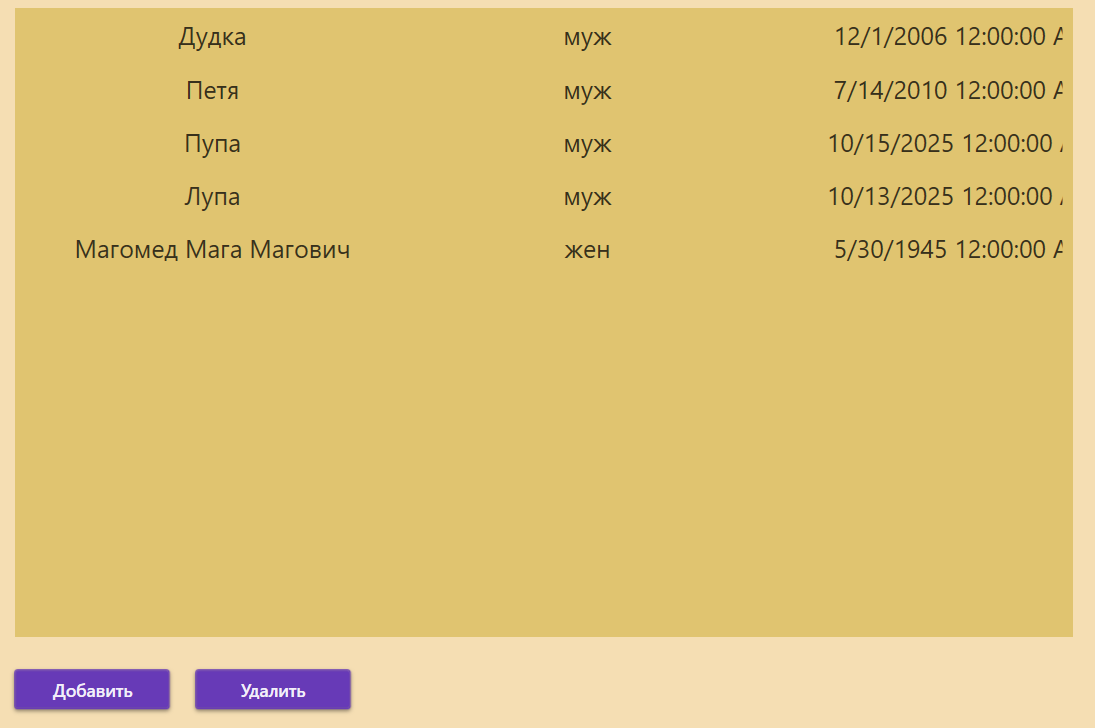


Рис.6 Окно пациентов, отображение всех вызовов

Для того чтобы удалить запись из таблицы нужно нажать на кнопку Удалить

Для того чтобы добавить запись в таблицу нужно нажать на кнопку Добавить и появится окно для добавления и изменения записи

Также стоит отметить, что редактирование записей реализовано двойным щелчком по элементу.



Рис.7 Редактирование пациентов

Это окно покупателей лотов в этом окне происходит редактирование всех покупателей и взаимодействие с этими объектами в БД (создание /удаление/изменение). Здесь можно изменить имя, логин и пароль. При нажатии кнопки сохранить запись сохраняется в БД. Если дважды нажать на поле из списка то можно будет изменить его и все изменения сохранятся в БД

В этом окне происходит инициализация всех докторов

Здесь используются конверторы для более понятной информации

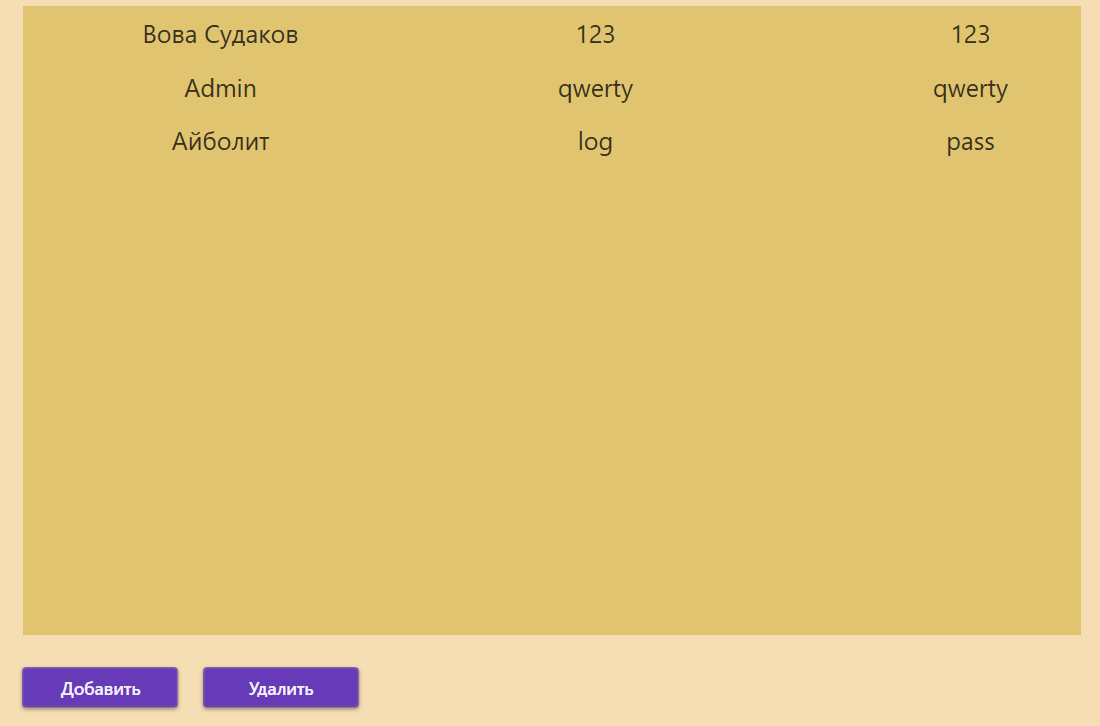


Рис.11 Окно докторов, отображение всех докторов

Для того чтобы удалить запись из таблицы нужно нажать на кнопку Удалить

Для того чтобы добавить запись в таблицу нужно нажать на кнопку Добавить и появится окно для добавления и изменения записи

Также стоит отметить, что редактирование записей реализовано двойным щелчком по элементу.

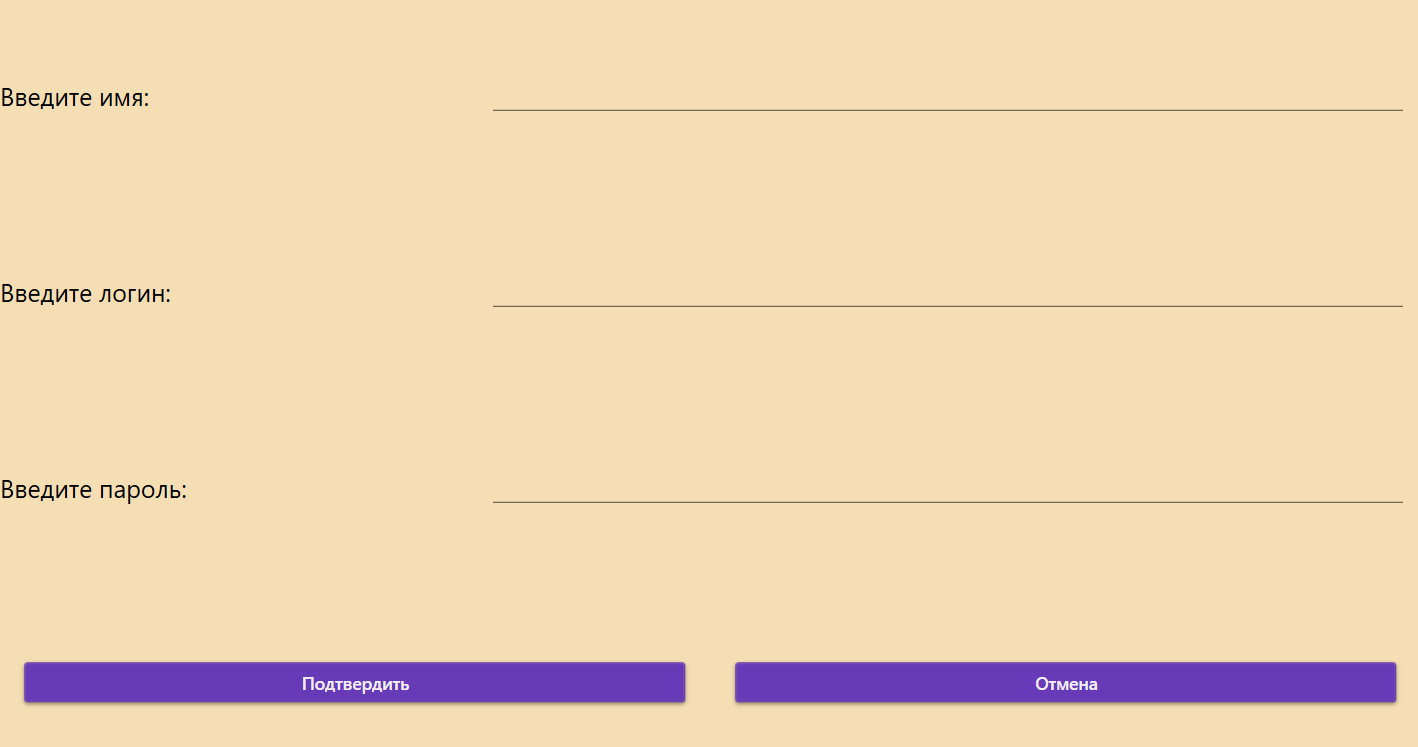


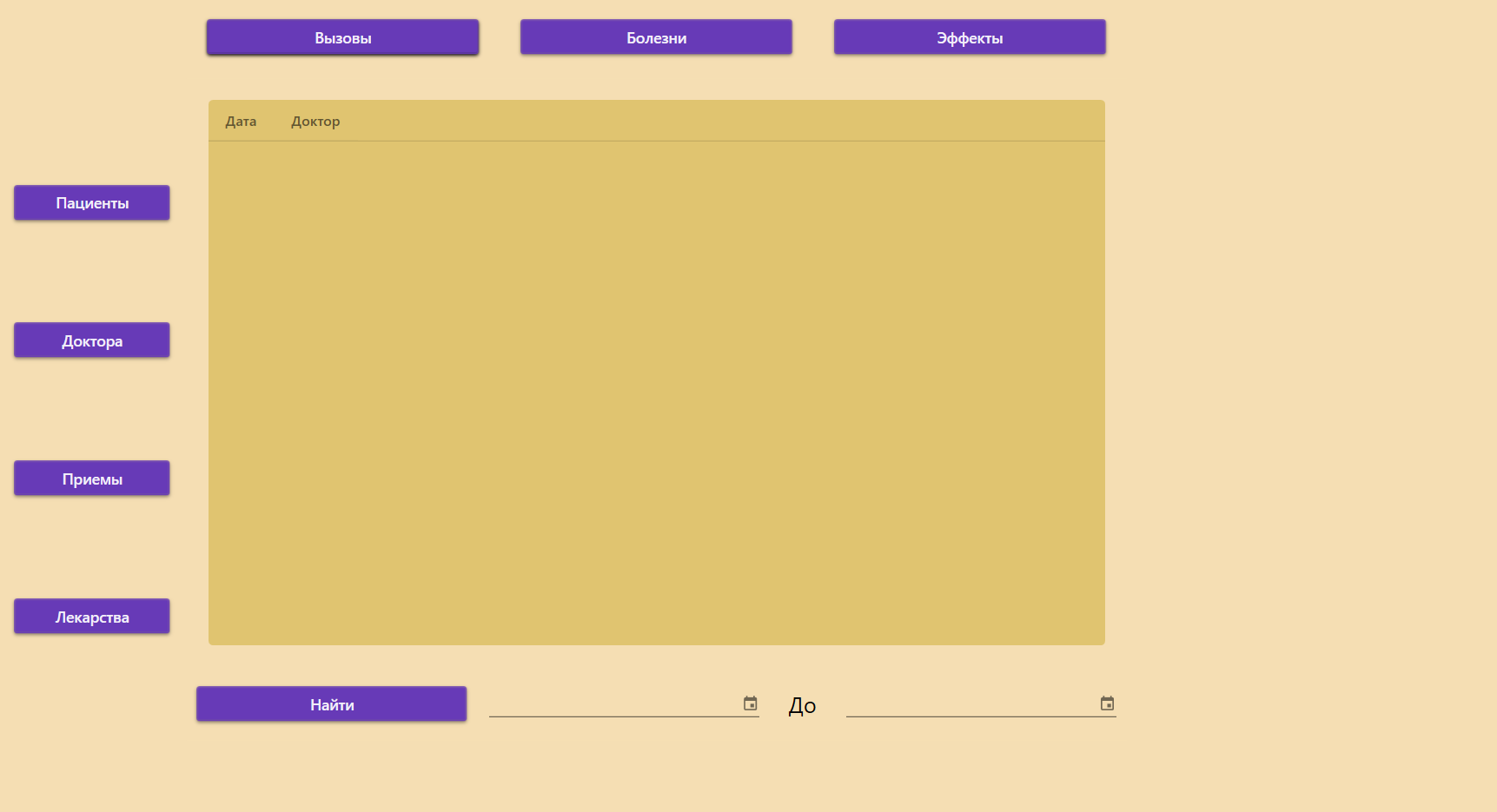
Рис.12 Редактирование докторов

Это окно добавления продавцов в этом окне происходит редактирование всех продавцов и взаимодействие с этими объектами в БД (создание /удаление/изменение). Здесь можно изменить имя, логин и пароль. При нажатии кнопки сохранить запись сохраняется в БД. Если дважды нажать на поле из списка то можно будет изменить его и все изменения сохранятся в БД

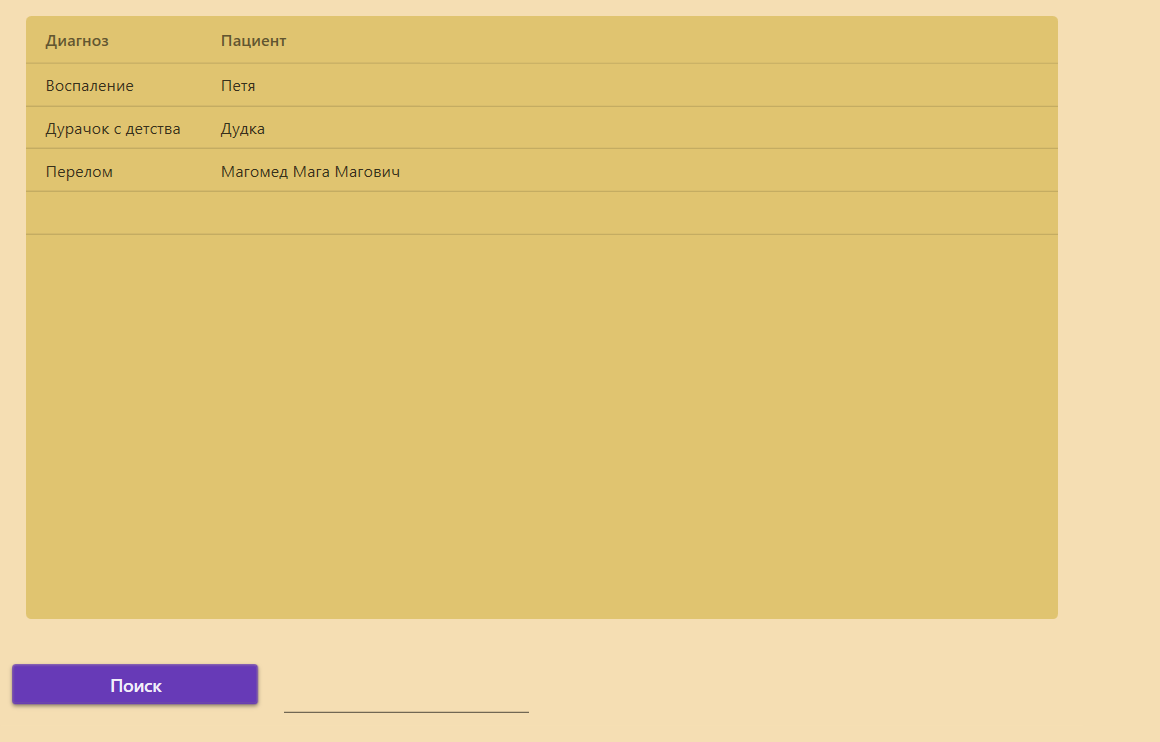
# Основные задачи



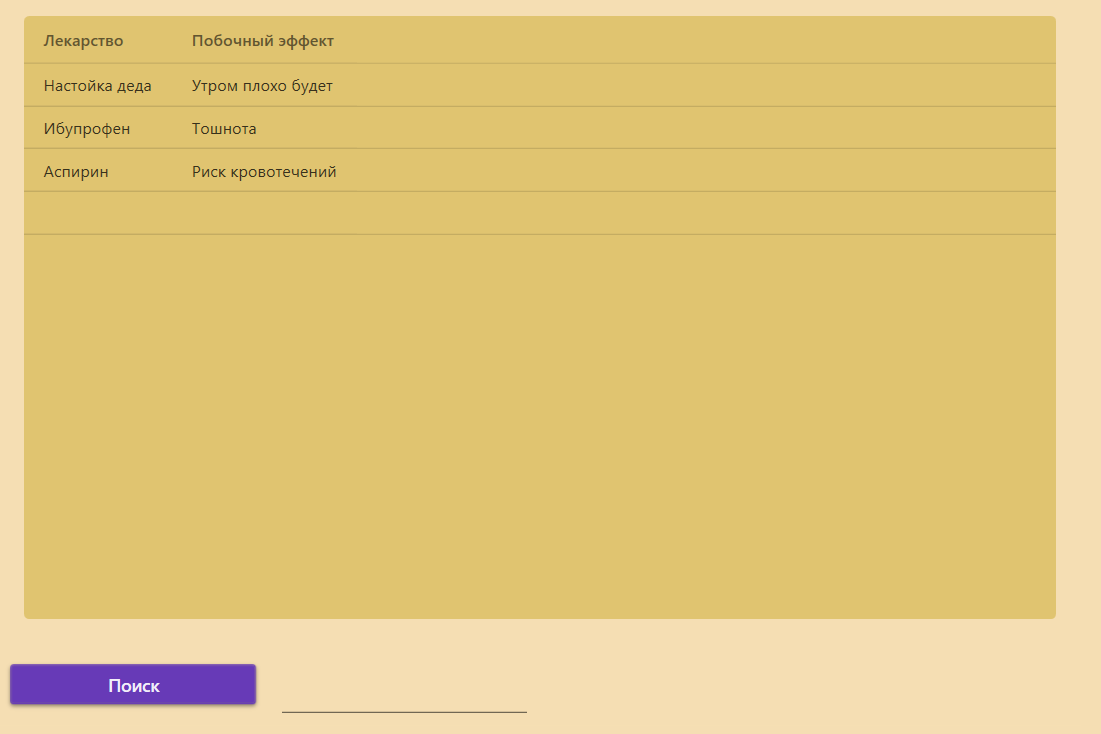
В этом окне происходит отображены основные задачи такие как Поиск вызовов, количество заболевших и побочные эффекты лекарств.



Здесь отображаются вызовы докторов в определенный интервал дат



Здесь отображается список заболевших



Здесь отображаются побочные эффекты от лекарств

# 4. Заключение

В ходе выполнения курсовой работы была успешно разработана [краткое описание проекта, например: система управления библиотекой, приложение для отслеживания финансов, и т.д.], демонстрирующая практическое применение ряда современных технологий разработки программного обеспечения. Работа охватила полный цикл создания проекта, от проектирования базы данных до разработки пользовательского интерфейса.

Центральное место в архитектуре приложения занял паттерн проектирования Model-View-ViewModel (MVVM), в сочетании с технологиями WPF (Windows Presentation Foundation) для построения пользовательского интерфейса и C# в качестве языка программирования. Применение MVVM позволило обеспечить четкое разделение модели данных (Model), логики представления (ViewModel) и пользовательского интерфейса (View), что существенно повысило модульность, тестируемость и поддерживаемость кода. Использование привязки данных в WPF упростило синхронизацию данных между моделью и представлением, сделав интерфейс более отзывчивым и интуитивно понятным.

Для работы с данными была выбрана система управления базами данных SQLite, практическое освоение которой позволило получить опыт проектирования и реализации реляционных баз данных, включая определение сущностей, атрибутов, связей и индексов. [Если использовался ORM, например Entity Framework, то добавить здесь]: Взаимодействие с базой данных осуществлялось с помощью Entity Framework, что позволило абстрагироваться от деталей работы с SQL и повысить производительность разработки.

В результате выполненной работы был получен [описание результатов – что именно создано и какие задачи решены]: рабочий прототип приложения с интуитивным пользовательским интерфейсом, эффективной архитектурой и надежной базой данных. Опыт, полученный в процессе разработки, позволил углубить понимание принципов объектно-ориентированного программирования, работы с базами данных и построения сложных приложений на основе паттерна MVVM. Полученные навыки позволят эффективно использовать WPF, C#, Entity Framework и SQLite в будущих проектах.

**Список литературы**

<https://metanit.com/sharp/>

<https://metanit.com/sharp/forms.php>

<https://metanit.com/sharp/ado.php>

<https://djvu.online/file/V7LO1eIy1qP70?ysclid=m4sxmmdub914998309>

<https://djvu.online/file/zjhb63g3Cz7yG?ysclid=m4sxo38lnb714222473>

<https://www.labirint.ru/books/846543/>

<https://vk.com/wall-51126445_103156?ysclid=m4sxsaclru849251668>