Министерство науки и образования Российской Федерации
Муромский институт (филиал)
Федерального государственного бюджетного образовательного учреждения
высшего образования
«Владимирский государственный университет
имени Александра Григорьевича и Николая Григорьевича Столетовых»
(МИ (филиал) ВлГУ)

Факультет ИТР

Кафедра ФПМ

Курсовая работа

по Базам данных.

Тема АС аптеки

Руководитель

Колпаков А.А,
(фамилия, инициалы)

(оценка)

(подпись) (дата)

Члены комиссии

Орлов А.А.
(подпись) (Ф.И.О)

(подпись) (Ф.И.О)

Студент ИБ-122
(группа)

Федоровский А.В.
(фамилия, инициалы)

(подпись) (дата)

Муром 2024

В работе представлена информационная система аптеки. Приложение разработано на платформе Windows. Цель работы – рассмотреть принципы разработки информационных систем и разработать программу на основе этих принципов.

Для разработки были выбраны язык программирования – C# и среда разработки – Visual Studio. Для создания базы данных использовалась СУБД SQLite.

В разработанном приложении доступны возомжность добавления, удаления, изменения и поиска записей в базе данных. Проведены тестирование и отладка программы.

The work presents an informational system of a pharmacy. The application is developed on the Windows platform. Purpose of the work is to consider the principles of development of information systems and develop a program, based on these principles

For development, programming language c# and development environment Visual Studio were chosen. The SQLite DBMS was used to create the database.

The developed application has the ability to add, delete, change and search for records in the database. Testing and debugging of program were carried out.

Содержание

Введение.....5
1.Анализ технического задания.....6
2.Разработка моделей данных.....7
2.1 Концептуальная модель.....7
2.2 Логическая модель.....9
2.3 Физическая модель.....10
3.Разработка и реализация ИС.....12
3.1 Создание базы данных.....13
3.2 Программирование ИС.....16
3.3 Работа над графическим приложением.....20
3.4 Руководство пользователя.....27
3.5 Руководство администратора.....30
3.6 Подробное описание программы.....31
4. Тестирование АИС.....33
Заключение.....38
Список литературы.....39
Приложение 1. Модели данных.....40
Приложение 2. Текст программы.....43
Приложение 3. Снимки окон программы.....41

Введение

Современные аптеки играют ключевую роль в системе здравоохранения, являясь посредниками между производителями и потребителями лекарств и различных медицинских приспособлений. Эффективное управление аптечными запасами и услугами требует применения современных информационных технологий. Для этого необходима надежная и функциональная база данных, которая позволит оптимизировать процессы учета, хранения и распространения медикаментов.

В данной курсовой работе рассматривается разработка базы данных для аптеки, которая будет обеспечивать автоматизацию учета товаров, поставщиков, сотрудников, а также взаимодействие с клиентами. Основное внимание уделяется структуре базы данных, её проектированию, что позволит не только упростить повседневные операции, но и повысить качество обслуживания клиентов.

Целью курсовой работы является создание информационной системы для добавления, удаления, изменения и поиска записей в базе данных.

Задачи, работающие для этой цели:

1. Проектирование концептуальной, логической и физической моделей базы данных.
2. Реализация физической модели в одной из современных СУБД.
3. Подключение БД к программе с помощью средств среды разработки.
4. Написание обработчиков событий и функций для добавления, удаления, изменения и поиска записей в БД.
5. Тестирование и редактирование

1. Анализ технического задания

Необходимо разработать автоматизированную систему (АС) для аптеки, которая будет иметь в себе функции хранения, изменения, добавления необходимых данных. Необходимо сделать приложение удобным и эффективным инструментом как в руках администратора аптеки, так и в руках обычного фармацевта, который сможет с помощью данного приложения более эффективно работать.

На данный момент на рынке находятся следующие продукты, подходящие под поставленную задачу:

1) Росконтроль: ПО для аптек с интеграцией государственных баз данных.
2) Medoc: система, специализированная на медикаментах, включая работу с рецептурой и анализом операций.
3) АБС: система, специализирующаяся на финансовый операциях.
4) Бейс: система, позволяющая делать автоматизацию аптек с учетом по товарам и клиникам.

В результате данной курсовой работы необходимо разработать приложение, объединяющее в себе удобство и функционал лучших продуктов на рынке.

Для правильного функционирования следует учесть следующие критерии:

- Наличие данных о сотрудниках, лекарствах, поставщиках и продажах.
- Возможность добавления, редактирования, поиска и удаления записей
- Наличие графического интерфейса, отображающего элементы базы данных

2. Разработка моделей данных

Разработка моделей данных представляет собой процесс проектирования структуры данных, которая будет использоваться в приложении для хранения и организации информации в базе данных (БД). Эти модели определяют, как данные будут представлены, связаны между собой и как они будут доступны для приложения.

Роль разработки моделей данных в проектировании приложения администрирования БД невозможно переоценить. Вот несколько ключевых аспектов:

Определение структуры БД: Разработка модели данных включает в себя определение сущностей (таких как таблицы) и их атрибутов.

Связи и нормализация: Разработка моделей данных помогает определить связи между таблицами, что важно для поддержания целостности данных. Применение нормализации данных (разделение таблиц для предотвращения избыточности) также является частью этого процесса.

Оптимизация запросов: Эффективные модели данных могут значительно повлиять на производительность запросов к базе данных. Правильное проектирование позволяет создать оптимизированные структуры данных, ускоряющие выполнение запросов.

2.1 Концептуальная модель

Создание концептуальной модели данных - это первый и абстрактный этап проектирования базы данных, который ориентирован на выделение основных сущностей и их взаимосвязей без углубленного внимания к техническим деталям.

Атрибуты - это конкретные данные, которые мы хотим сохранить для каждой сущности. Связи отражают, как сущности взаимодействуют друг с другом. Например, чек связан с клиентом и лекарствами. Определение типа связей (один-к-одному, один-ко-многим, многие-ко-многим) важно для правильного построения модели.

Будем стремиться к минимализации избыточности данных, избегая повторения между сущностями для сохранения целостности и согласованности данных

Визуализация концептуальной модели происходит чаще всего с использованием диаграммы сущность-связь (ER-диаграммы). Это графическое представление сущностей, их атрибутов и связей между ними.

Так как аптека в первую очередь работает с различными лекарствами, то следовательно в нашей базе данных должна быть сущность лекарства с атрибутами: цена, единица измерения, доступное количество, название, группа, продажа по рецепту и поставщик, который в свою очередь, то же является сущностью с атрибутами: название, ФИО агента, номер телефона, адрес. По мимо этого выделим отдельную сущность покупка, с атрибутом количество и сумма, которая будет содержать купленные лекарства, данная сущность нужна для соединения между собой покупателя и лекарств в чеке. Так же в нашей ИС будут содержаться сведения о сотрудниках аптеки, которые там работают с атрибутами: ФИО, должность, паспорт, дата рождения, дата найма, телефон, график работы, зарплата, адрес, логин и пароль для входа в систему. Сведения о покупателях с атрибутами ФИО, номер телефона, паспорт, адрес. а так же сущность Чек с атрибутами дата продажи и сумма.

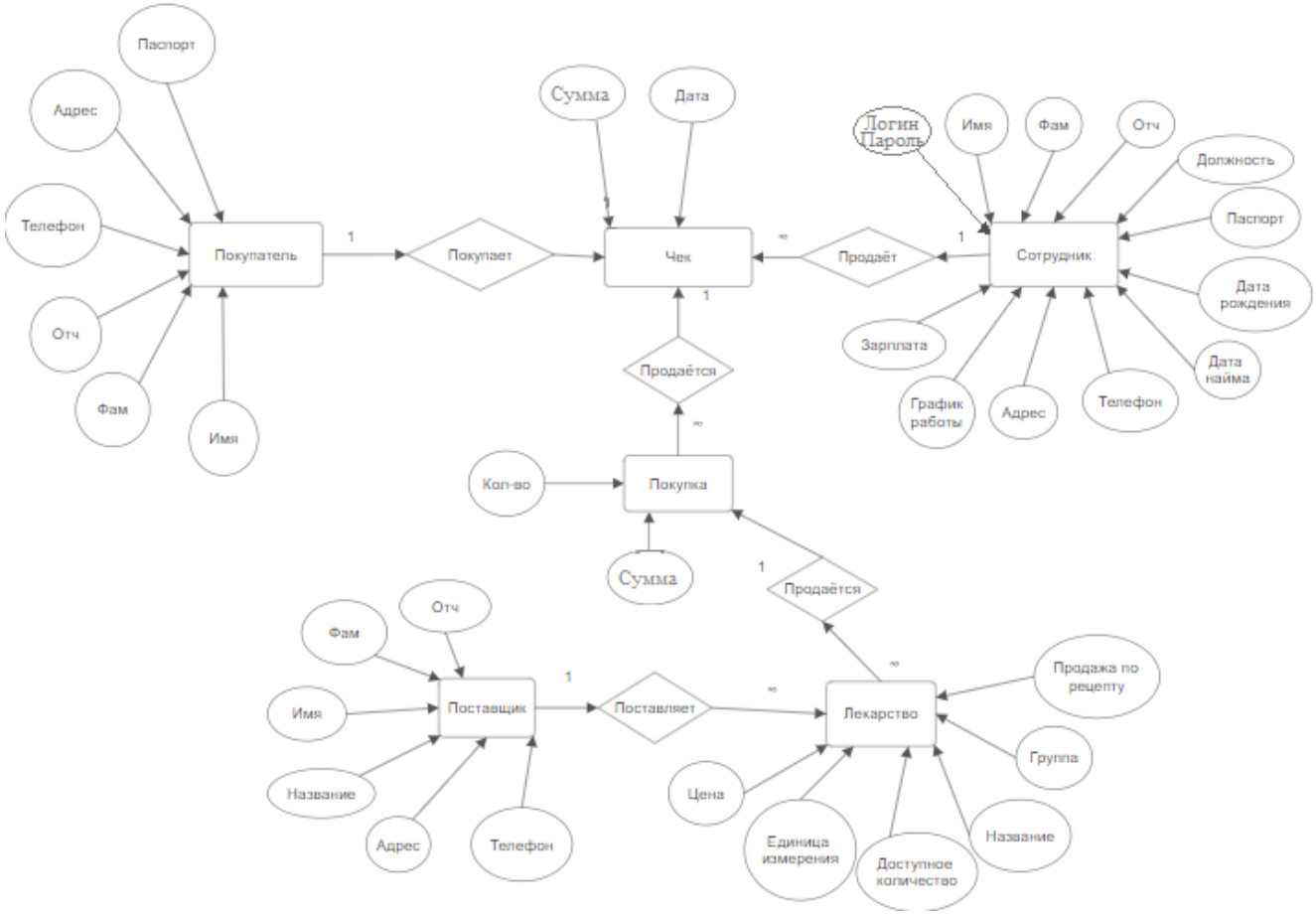


Рисунок 1: Концептуальная модель данных

Таким способом, в нашей базе данных будут следующие отношения между сущностями: Поставщик поставляет лекарства, которые продавец продаёт покупателю, а покупатель, соответственно, их покупает, всё это отраженно в сущности Чек с соответствующими сложными атрибутами: покупатель, продавец и лекарства.

2.2 Логическая модель

Логическая модель базы данных представляет собой более конкретное и техническое описание того, как данные будут организованы и храниться в базе данных. Каждая сущность в концептуальной модели становится таблицей в логической модели. Каждая запись в таблице представляет собой конкретный экземпляр сущности. Следует отметить, что для избежания каких-либо изменений ифнормации в таблицах «чек» и «покупка», было принято решение убрать от туда вторичные ключи и использовать записи с такими же именнами для сохранения данных в неизменном состоянии. Для этой же цели в таблицах «Чек» и «Покупка» были объявленны атрибуты «сумма».

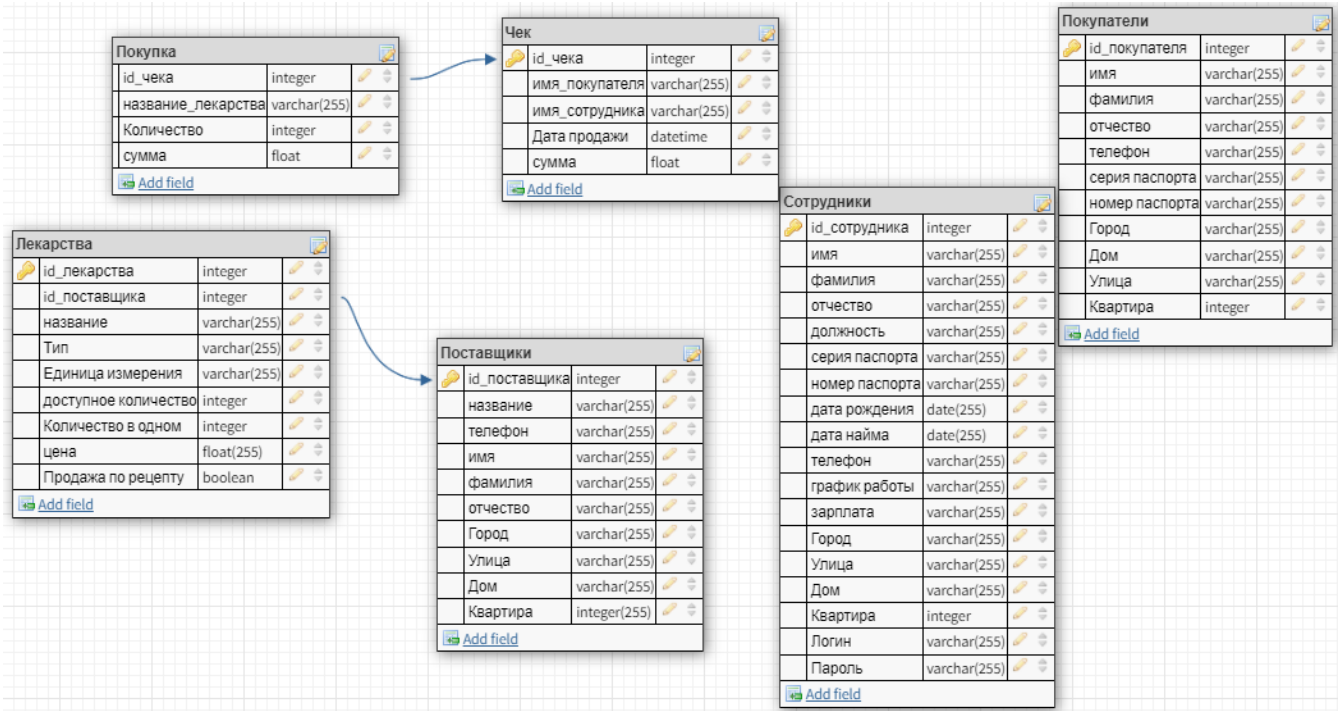


Рисунок 2: Логическая модель данных

Логическая модель базы данных является промежуточным звеном между концептуальной моделью и физической реализацией базы данных. Она предоставляет детальное понимание структуры данных, не зависящее от конкретной СУБД, что позволяет легче адаптировать модель под разные технологические решения.

2.3 Физическая модель

Физическая модель базы данных представляет собой конкретную реализацию структуры данных на уровне конкретной СУБД. Эта модель включает в себя информацию о том, как данные будут храниться, какие индексы будут использованы, и другие технические аспекты. Как видно из рисунка 3, в физической модели мы определили типы данных, которые будут использоваться в нашей базе данных в соотвествии с типами данных, используемыми в SQLite. Так, например, булевое значение «продажа по рецепту» из таблицы «Лекарства» будет хранится в строке с дополнителльной проверкой на введённые значения.

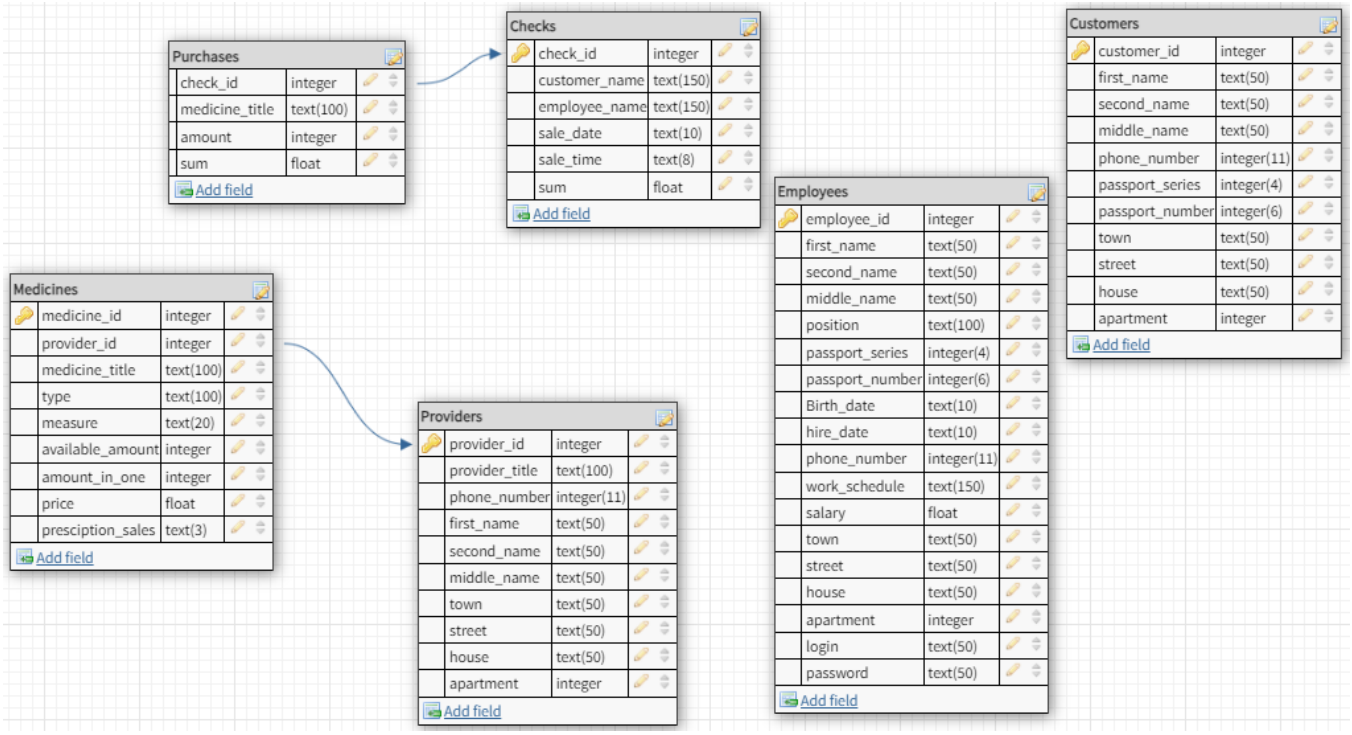


Рисунок 3: Физическая модель данных

3. Разработка и реализация ИС

Автоматизированная система (АС) представляет собой комплекс взаимосвязанных программных и аппаратных средств, ориентированных на сбор, обработку, хранение и передачу информации внутри организации.

В автоматизированных информационных системах за хранение любой информации отвечают:

1. На физическом уровне:

а. внешние накопители;

б. встроенные устройства памяти (RAM);

в. массивы дисков;

2. На программном уровне:

а. СУБД;

б. файловая система ОС;

в. системы хранения мультимедиа, документов и т. д.

Информационная система (ИС) в данном контексте представляет собой программный комплекс, который обеспечивает сбор, обработку, хранение и предоставление доступа к данным о персонале, клиентах, поставщиках и лекарствах. Наше приложение будет являться частью этой ИС, предоставляя возможности по удобному взаимодействию с данными аптеки. Оно позволит эффективно управлять организацией, автоматизировать рутинные задачи и предоставит инструменты для анализа данных, содействуя более эффективному принятию решений по организации аптечного дела.

Для эффективной разработки программы, взаимодействующей с базой данных, будем применять объектно-ориентированный подход с использованием языка программирования C# и графического модуля windows forms.

В контексте создания графического интерфейса приложения, C# предлагает множество преимуществ:

Основной фокус при разработке должен быть направлен на создание удобного и интуитивно понятного интерфейса. C# обеспечивает простоту и

гибкость в этом плане, позволяя легко реализовывать все необходимые элементы управления для оптимизации пользовательского опыта.

Надежная обработка данных также лежит в основе разработки. C# обеспечивает эффективную работу с данными, гарантируя точность и своевременность вывода результатов пользователю. Благодаря своей отзывчивости на действия пользователя, приложение на C# способно обеспечивать мгновенный отклик и формировать выходные данные в соответствии с запросами пользователей.

Согласно выдвинутым требованиям к приложению база данных должна содержат в себе несколько таблиц, связанных между собой. Для разработки базы данных была выбрана среда SQLite, которая является СУБД реляционного вида.

К преимуществам SQLite относится:

- Простота в использовании - SQLite не требует установки отдельного сервера, так что его легко интегрировать в проект.
- Быстрая реализация - Благодаря своей простоте и легкости использования, SQLite позволяет быстро начать работу над проектом, минимизируя время разработки.
- Кроссплатформенность - SQLite поддерживает множество платформ, что обеспечивает кроссплатформенность приложений.
- Широкий круг поддерживаемых данных - SQLite поддерживает разнообразные типы данных, включая текстовые, числовые, временные и бинарные данные.

3.1 Создание базы данных

Перед тем, как связать таблицы, их необходимо создать, воспользуемся физической моделью базы данных и создадим на её основе реальную базу данных в SQLite:

AptekaDB releas Имя таблицы: Checks WITHOUT ROWID S

	Имя	Тип данных	Первичный ключ	Внешний ключ	Уникальность	Проверка	Не NULL	Срав
1	check_id	INTEGER						
2	customer_id	INTEGER						
3	employee_id	INTEGER						
4	sale_date	TEXT (10)						
5	sale_time	TEXT (8)						
6	sum	NUMERIC						

Рисунок 4: Таблица «Чеки»

AptekaDB releas Имя таблицы: Purchases WITHOUT ROWID

	Имя	Тип данных	Первичный ключ	Внешний ключ	Уникальность	Проверка	Не NULL	Сра
1	check_id	INTEGER						
2	medicine_id	INTEGER						
3	amount	INTEGER						
4	sum	NUMERIC						

Рисунок 5: Таблица «Покупки»

AptekaDB releas Имя таблицы: Customers WITHOUT ROWID STRICT

	Имя	Тип данных	Первичный ключ	Внешний ключ	Уникальность	Проверка	Не NULL	Срав
1	customer_id	INTEGER						
2	first_name	TEXT (50)						
3	second_name	TEXT (50)						
4	middle_name	TEXT (50)						
5	phone_number	INTEGER (11)						
6	passport_series	INTEGER (4)						
7	passport_number	INTEGER (6)						
8	town	TEXT (50)						
9	street	TEXT (50)						
10	house	TEXT (50)						
11	apartment	INTEGER						

Рисунок 6: Таблица «Покупатели»

AptekaDB releas Имя таблицы: Employees ☐ WITHOUT ROWID ☐ STRICT

	Имя	Тип данных	Первичный ключ	Внешний ключ	Уникальность	Проверка	Не NULL	Сравн
1	employee_id	INTEGER						
2	first_name	TEXT (50)						
3	second_name	TEXT (50)						
4	middle_name	TEXT (50)						
5	position	TEXT (100)						
6	passport_series	INTEGER (4)						
7	passport_number	INTEGER (6)						
8	birth_date	TEXT (10)						
9	hire_date	TEXT (10)						
10	phone_number	INTEGER (11)						
11	work_schedule	TEXT (150)						
12	salary	NUMERIC						
13	town	TEXT (50)						

Рисунок 7: Таблица «Сотрудники» ч.1

AptekaDB releas Имя таблицы: Employees ☐ WITHOUT ROWID ☐ STRIC

	Имя	Тип данных	Первичный ключ	Внешний ключ	Уникальность	Проверка	Не NULL	Сра
6	passport_series	INTEGER (4)						
7	passport_number	INTEGER (6)						
8	birth_date	TEXT (10)						
9	hire_date	TEXT (10)						
10	phone_number	INTEGER (11)						
11	work_schedule	TEXT (150)						
12	salary	NUMERIC						
13	town	TEXT (50)						
14	street	TEXT (50)						
15	house	TEXT (50)						
16	apartment	INTEGER						
17	login	TEXT (50)						
18	password	TEXT (50)						

Рисунок 8: Таблица «Сотрудники» ч.2

AptekaDB releas Имя таблицы: Medicines WITHOUT ROWID STRIC

	Имя	Тип данных	Первичный ключ	Внешний ключ	Уникальность	Проверка	Не NULL	Сра
1	medicine_id	INTEGER						
2	provider_id	INTEGER						
3	medicine_title	TEXT (100)						
4	type	TEXT (100)						
5	measure	TEXT (20)						
6	amount_in_one	INTEGER						
7	available_amount	INTEGER						
8	price	NUMERIC						
9	presciption_sales	TEXT (3)						

Рисунок 9: Таблица «Лекарства»

AptekaDB releas Имя таблицы: Providers WITHOUT ROWID STRICT

	Имя	Тип данных	Первичный ключ	Внешний ключ	Уникальность	Проверка	Не NULL	Срав
1	provider_id	INTEGER						
2	provider_title	TEXT (100)						
3	phone_number	INTEGER (11)						
4	first_name	TEXT (50)						
5	second_name	TEXT (50)						
6	middle_name	TEXT (50)						
7	town	TEXT (50)						
8	street	TEXT (50)						
9	house	TEXT (50)						
10	apartment	INTEGER						

Рисунок 10: Таблица «Поставщики»

3.2 Программирование ИС

Для создания системы будет использована среда разработки Visual Studio и язык c#, с импортированным модулем System.Data.SQLite для работы с базами данных. База данных должна храниться в папке с программой.

Подключение к базе данных происходит при помощи метода SQLiteConnection и все последующие манипуляции с базой данных выполняются через него

```
private string connectionString = "Data Source=AptekaDB.db;Version=3;";
using (SQLiteConnection con = new SQLiteConnection(connectionString))
    { //Последующий код здесь}
```

Для вывода информации из базы данных используются заранее сгенерированные Select запросы Передающиеся в объект SQLiteDataAdapter.

```
private void SeeCustomers(string keyword = null, List<string> columnNames = null)
    {
        try
        {
            if (keyword == null)
            {
                using (SQLiteConnection con = new SQLiteConnection(connectionString))
                {
                    dataGridView1.DataSource = null;
                    con.Open();
                    SQLiteDataAdapter adapter = new SQLiteDataAdapter("select customer_id, first_name as имя, second_name as фамилия, middle_name as отчество, phone_number as номер_телефона, passport_series as серия_паспорта, passport_number as номер_паспорта, town as город, street as улица, house as дом, apartment as квартира from Customers", con);
                    DataTable dt = new DataTable();
                    adapter.Fill(dt);
                    dataGridView1.DataSource = dt;
                    label1.Text = "";
                }
            }
```

Поиск по таблице так же происходит при помощи данных запросов, модифицированных условием.

```
else
            {
                using (SQLiteConnection con = new SQLiteConnection(connectionString))
                {
                    dataGridView1.DataSource = null;
                    con.Open();
                    string script = "select customer_id, first_name as имя, second_name as фамилия, middle_name as отчество, phone_number as номер_телефона, " +
                        "passport_series as серия_паспорта, passport_number as номер_паспорта, town as город, street as улица, house as дом, apartment as квартира from Customers where ";
                    for (int i = 0; i < columnNames.Count; i++)
                    {
                        script += $"{columnNames[i]} LIKE '%{keyword}%'";
                        if (i < columnNames.Count - 1)
                        {
                            script += " OR "; // Добавляем OR между условиями
                        }
                    }
```

Для удаления записей из таблицы используется заранее сгенерированные delete запрос, в который передается имя таблицы и id записи, необходимой к удалению.

```
using (SQLiteConnection con = new SQLiteConnection(connectionString))
        {
            con.Open();
            var command = new SQLiteCommand($"DELETE FROM {table_name} WHERE {column_name} = {id}", con);
            command.ExecuteNonQuery();
            // Обновление DataGridView
            if (table_name == "Customers") SeeCustomers();
            // Ещё несколько условий для показа результата
```

Создание новых записей в таблицах происходи через специальные формы(уникальные для каждой таблицы), в которые пользователь заносит необходимые данные, после чего они помещаются в заранее сгенерированные insert запрос.

```
using (SQLiteConnection con = new SQLiteConnection(connectionString))
        {
            con.Open();
              string script = $"insert into Customers(first_name, second_name, middle_name, phone_number, passport_series, passport_number, town, street, house, apartment) values (\"{first_name}\", \"{second_name}\", \"{middle_name}\", \"{phone_number}\", \"{pas_ser}\", \"{pas_num}\", \"{town}\", \"{street}\", \"{house}\", \"{apartment}\")";
            using (SQLiteCommand command = new SQLiteCommand(script, con))
            {
                int rowsAffected = command.ExecuteNonQuery();
            }
            MessageBox.Show("Запись успешно добавлена");
            this.Close();
        }
```

Точно так же происходит изменение данных в таблице, пользователь вводит необходимые изменения в специальную форму, которые передаются в заранее сгенерированные update запрос.

```
using (SQLiteConnection con = new SQLiteConnection(connectionString))
        {
            con.Open();
            string script;
              script = $"update Customers set first_name = \"{first_name}\", second_name = \"{second_name}\", middle_name = \"{middle_name}\", phone_number = \"{phone_number}\", passport_series = \"{pas_ser}\", passport_number = \"{pas_num}\", town = \"{town}\", street = \"{street}\", house = \"{house}\", apartment = \"{apartment}\" where customer_id = \"{customer_id}\"";
            using (SQLiteCommand command = new SQLiteCommand(script, con))
            {
                int rowsAffected = command.ExecuteNonQuery();
```

```
            }
            MessageBox.Show("Запись успешно обновленна");
            this.Close();
        }
```

Для получения цены на лекарства при добавлении их в чек будет выполняться следующий запрос:

```
string script = $"SELECT (price * {amount}) as sum from Medicines m where medicine_id = \"{medicine_id}\"";
        using (SQLiteCommand command = new SQLiteCommand(script, connection))
        {
            object result = command.ExecuteScalar();
            sum = result.ToString();
        }
```

При закрытии чека будет заполнятся таблица покупки:

```
foreach (DataGridViewRow row in dataGridView1.Rows)
        {
            // Пропускаем пустые строки
            if (row.IsNewRow) continue;
            script = $"insert into Purchases(check_id, medicine_title, amount, sum) values (\"{check_id}\", \"{row.Cells[0].Value}\", \"{row.Cells[1].Value}\", \"{row.Cells[2].Value}\")";
            using (SQLiteCommand com = new SQLiteCommand(script, con))
            {
                com.ExecuteNonQuery();
            }
        }
```

И после этого будет уменьшаться количество товара на складе:

```
script = $"update Medicines set available_amount = available_amount - (select amount from Purchases where Purchases.medicine_title = Medicines.medicine_title and Purchases.check_id = {check_id}) where medicine_title in (select medicine_title from Purchases where check_id = {check_id})";
        using (SQLiteCommand command = new SQLiteCommand(script, con))
        {
            int rowsAffected = command.ExecuteNonQuery();
        }
```

Код для кнопок, показывающих таблицы, будет выглядеть следующим образом:

```
private void Customer_button_Click(object sender, EventArgs e)
    {
        SeeCustomers();
        Explore_check_button.Visible = false;
        change_button.Visible = true;
        delete_button.Visible = true;
    }
```

Код для кнопок в меню, предназначенный для добавления записей будет выглядеть следующим образом:

```
private void покупателитoolStripMenuItem1_Click(object sender, EventArgs e)
    {
      FormCustomerADD newform = new FormCustomerADD();
      newform.ShowDialog();
      SeeCustomers();
      Explore_check_button.Visible = false;
      change_button.Visible = true;
      delete_button.Visible = true;
    }
```

В некотрых формах для добавления и изменения записей будут использоваться комбобоксы (comboBox) для замены id поля на понятное пользователю название:

```
private void comboBox2_SelectedIndexChanged(object sender, EventArgs e)
    {
      var selectedItem = (dynamic)comboBox2.SelectedItem;
      medicine_id = Convert.ToString(selectedItem.Value);
      medicine_title = Convert.ToString(selectedItem.Text);
    }
```

3.3 Работа над графическим приложением

Приложение будет разделено на 2 части: для администратора и для сотрудника аптеки.

При запуске приложения пользователю будет предложено аутентифицироваться для дальнейшей работы:

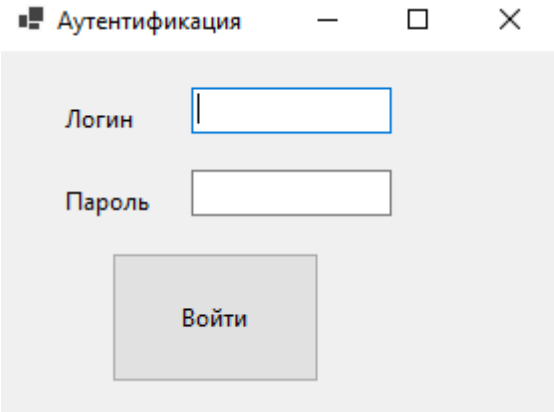


Рисунок 11: Окно аутентификации

У администратора будет собственная рабочая форма, в которой он сможет взаимодействовать с базой данных, а именно добавлять, изменять и удалять записи.

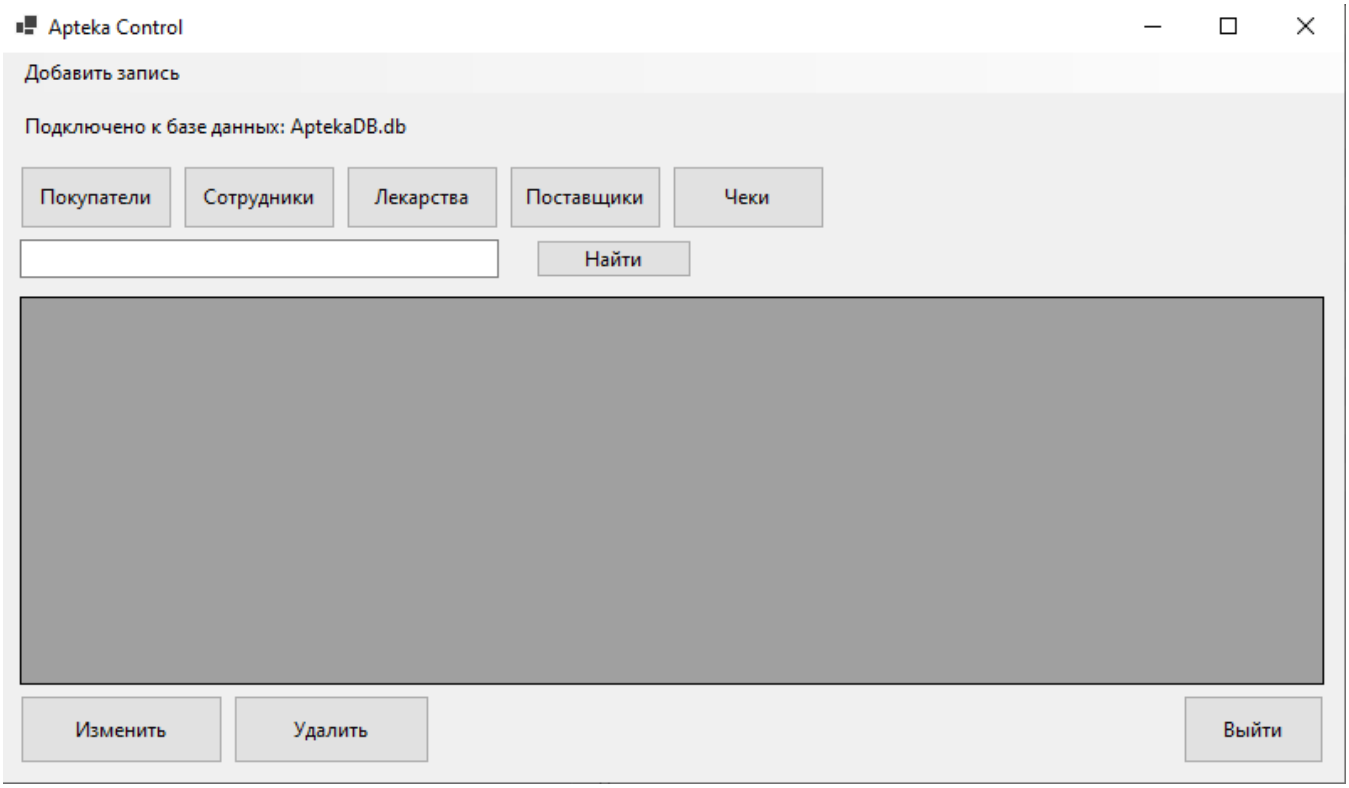


Рисунок 12: Рабочее пространство администратора

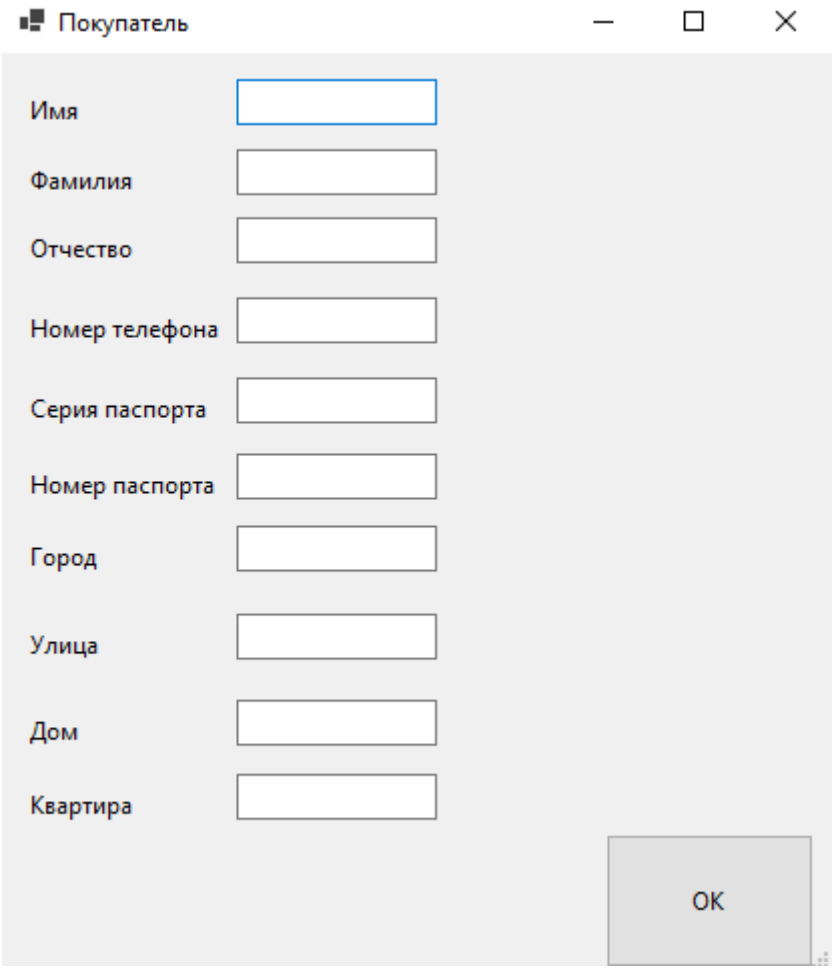


Рисунок 13: Окно для добавления и изменения записей о покупателях

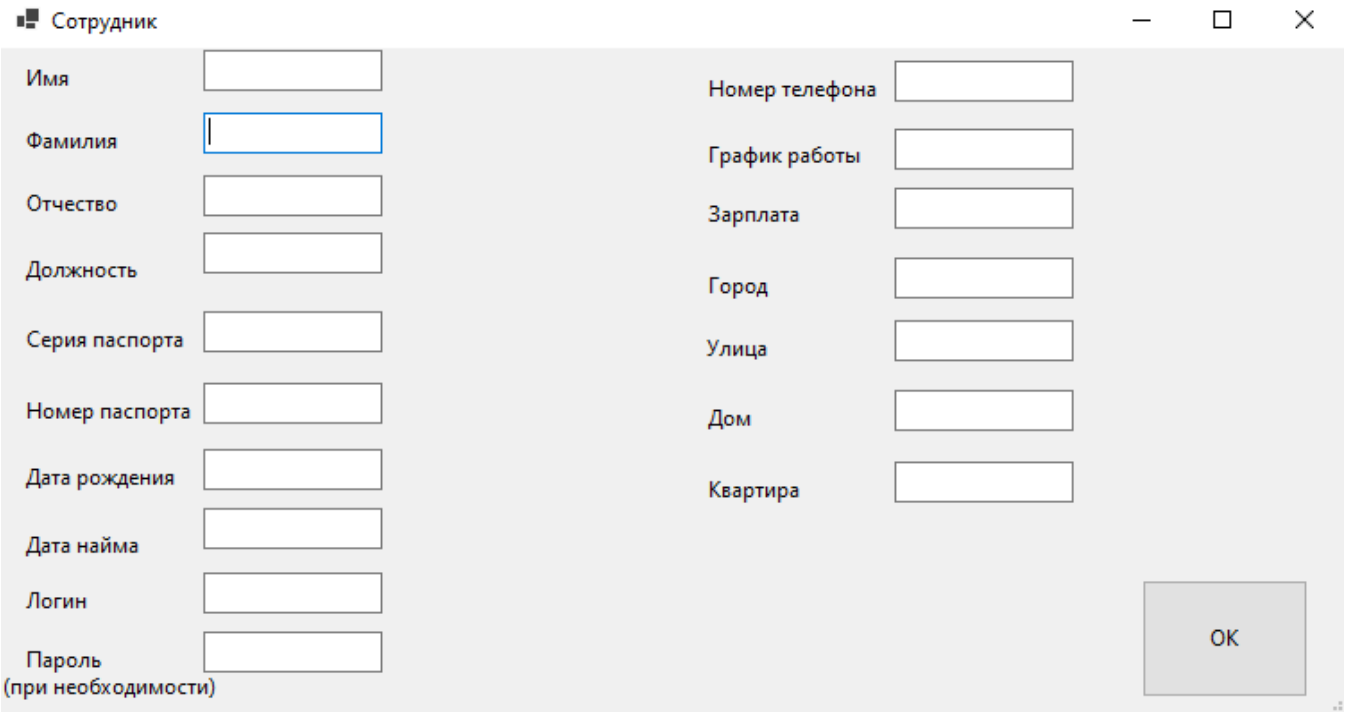


Рисунок 14: Окно для добавления и изменения записей о сотрудниках

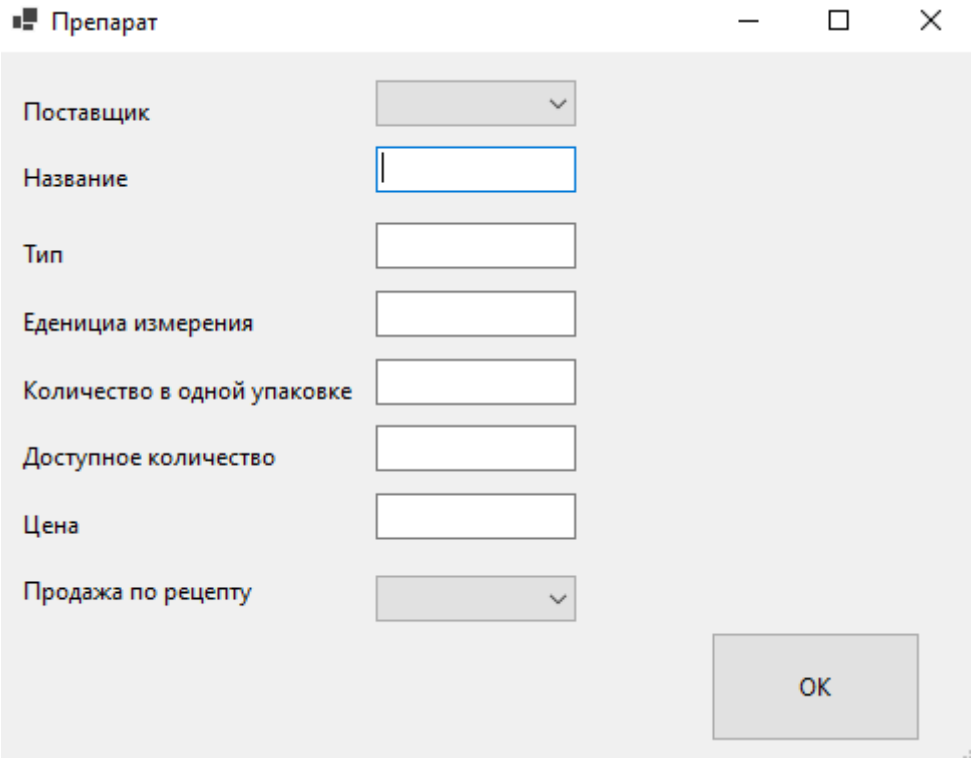


Рисунок 15: Окно для добавления и изменения записей о лекарствах

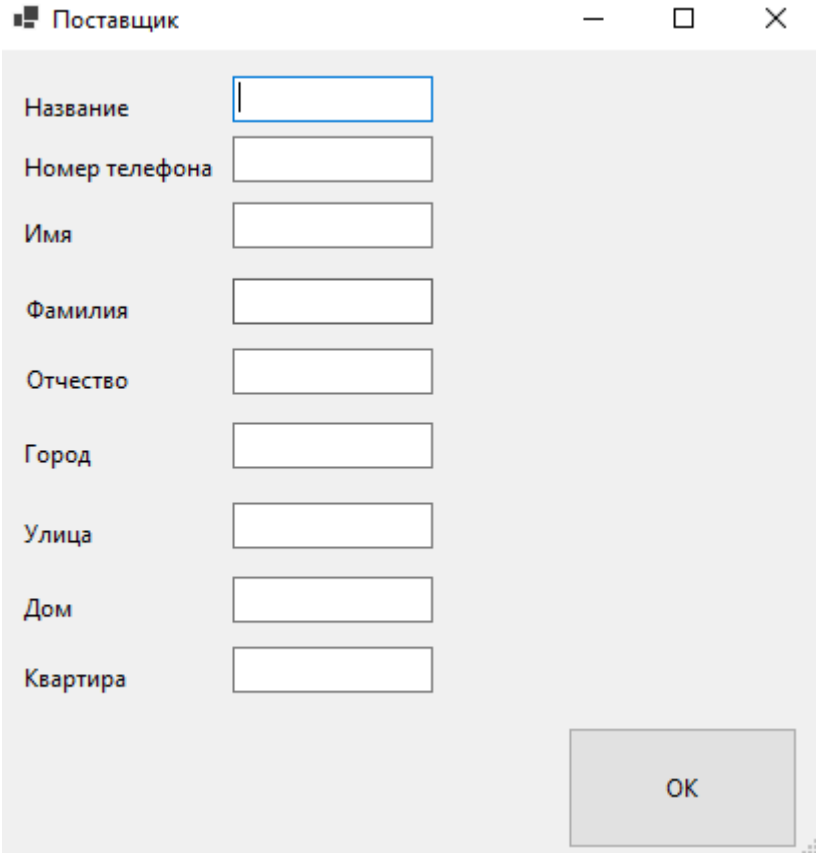


Рисунок 16: окно для добавления и изменения записей о поставщиках

В свою очередь, у сотрудников аптеки есть своё рабочее пространство, где они не могут редактировать все записи базы данных, но могут записывать проданные товары в чеки(чего не может делать администратор). У каждого

сотрудника есть своя форма, а так же логин и пароль для входа в неё, который хранится в базе данных в зашифрованном виде.

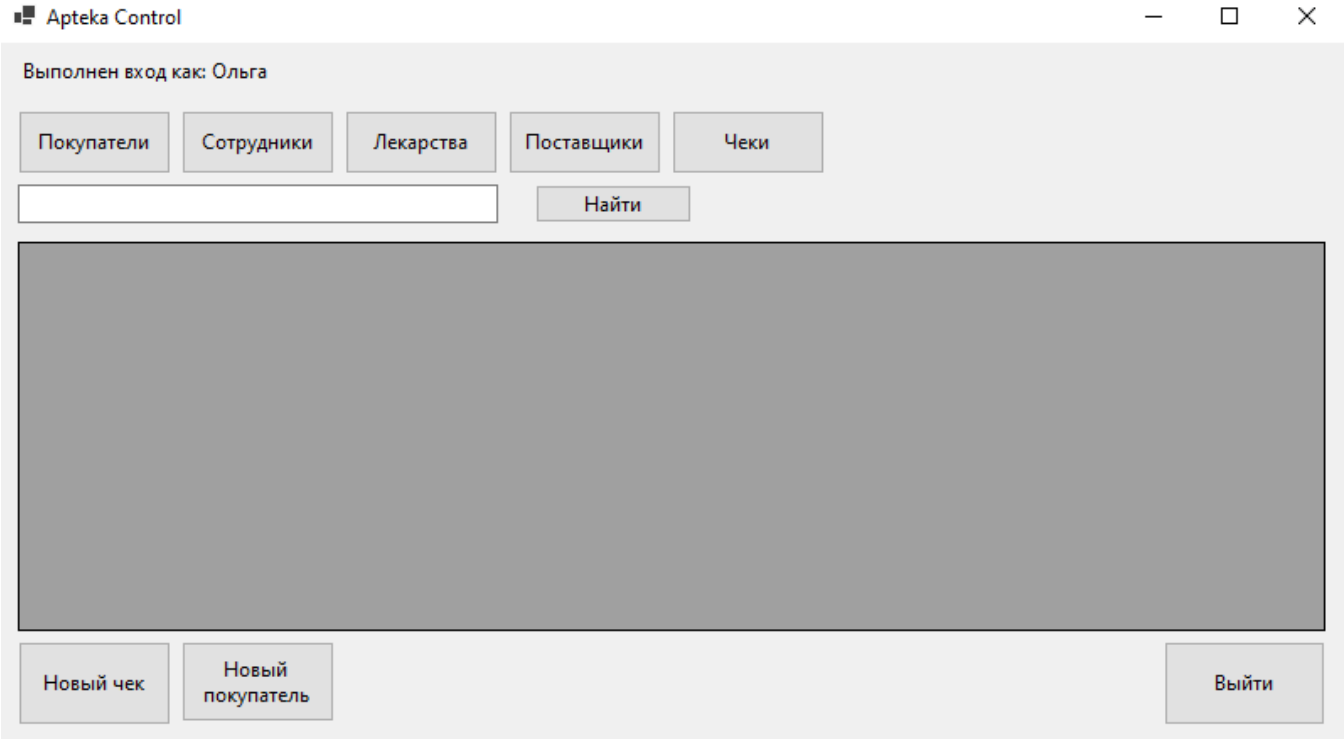


Рисунок 17: Рабочее пространстов сотрудника

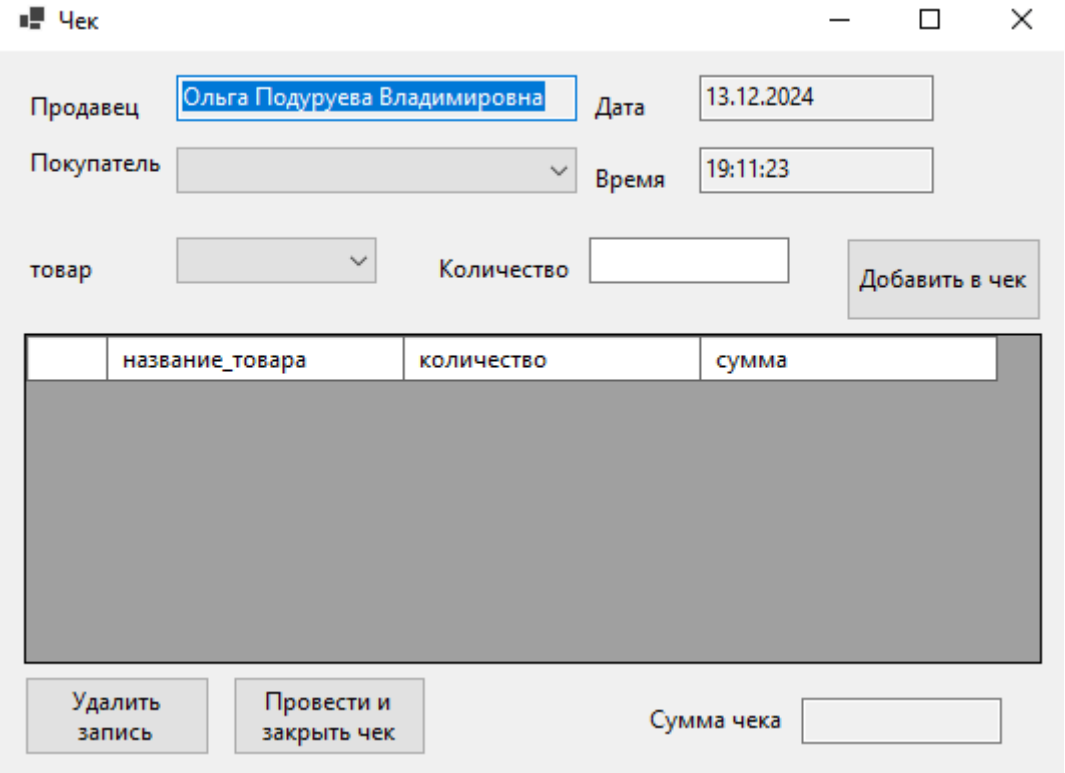


Рисунок 18: Окно для добавления нового чека

Администратор и фармацевт могут просматривать записи в базе днных через соответствующие кнопки.

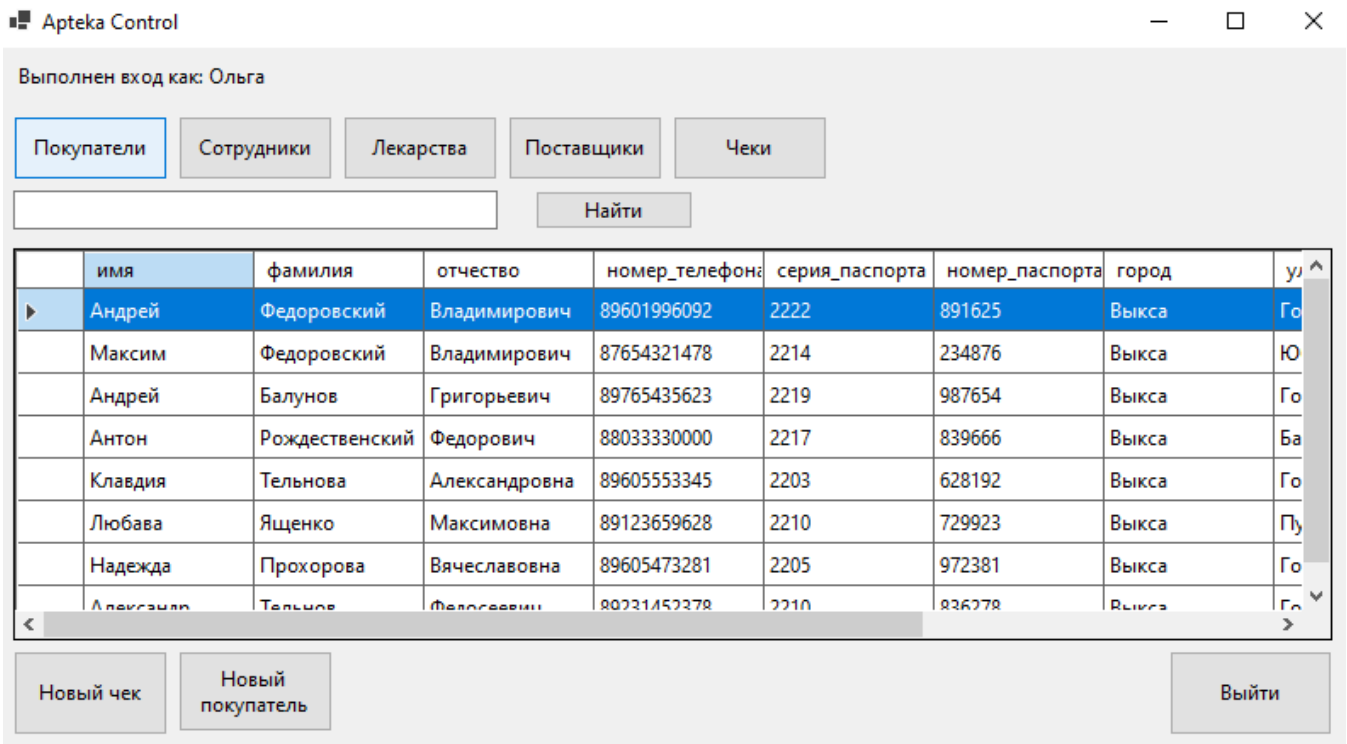


Рисунок 19: таблица «Покупатели»

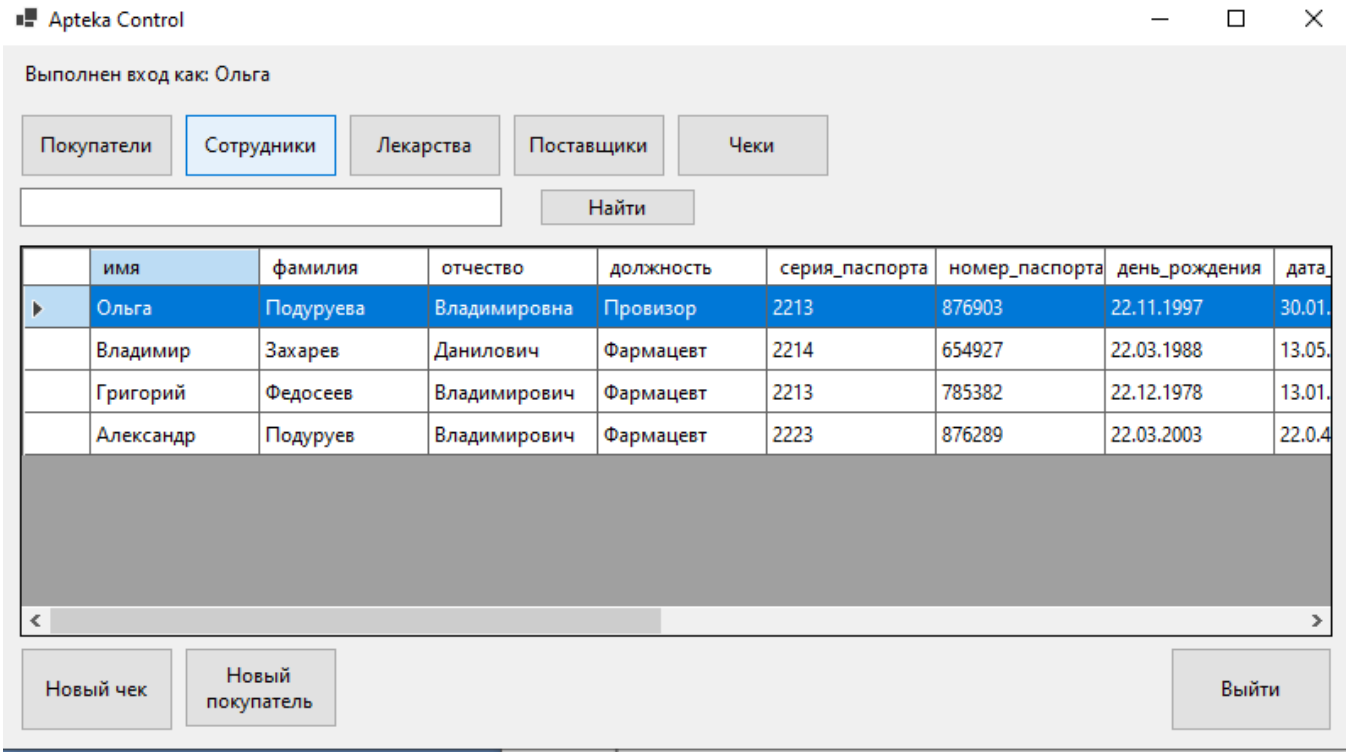
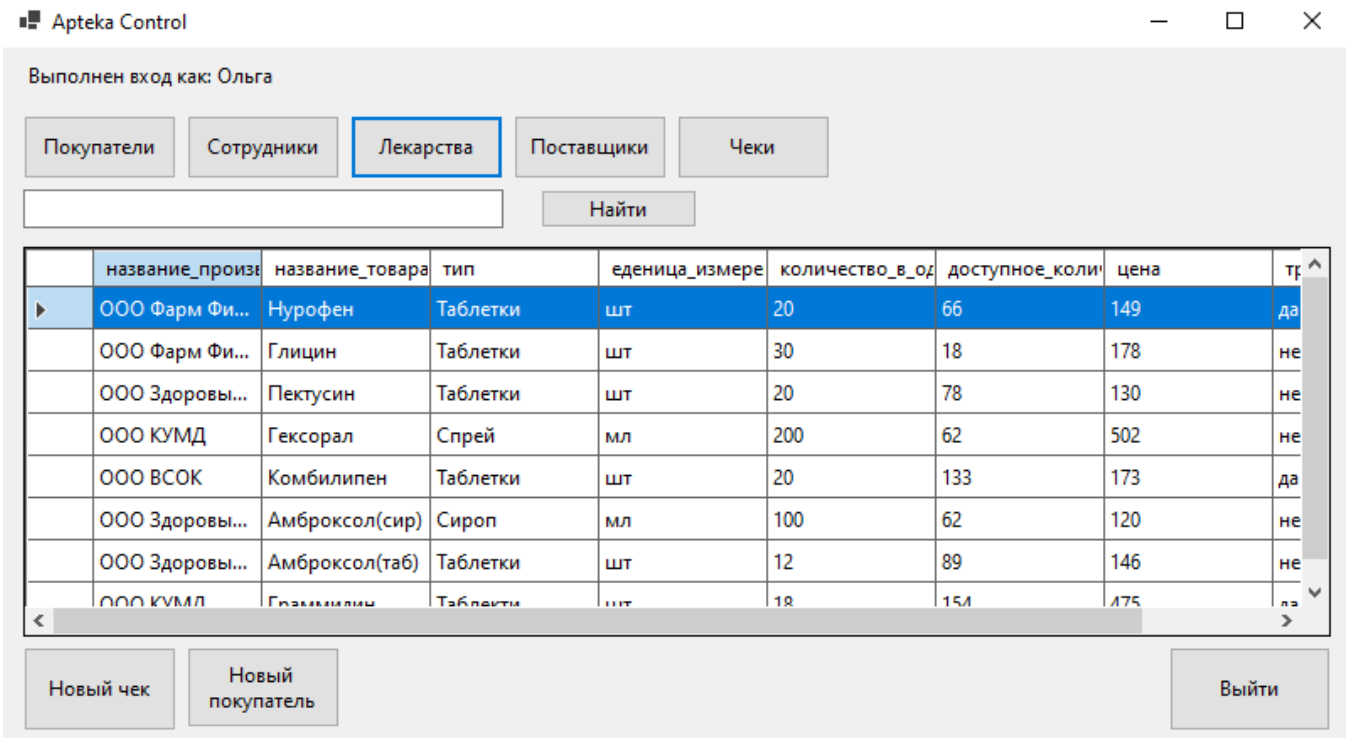
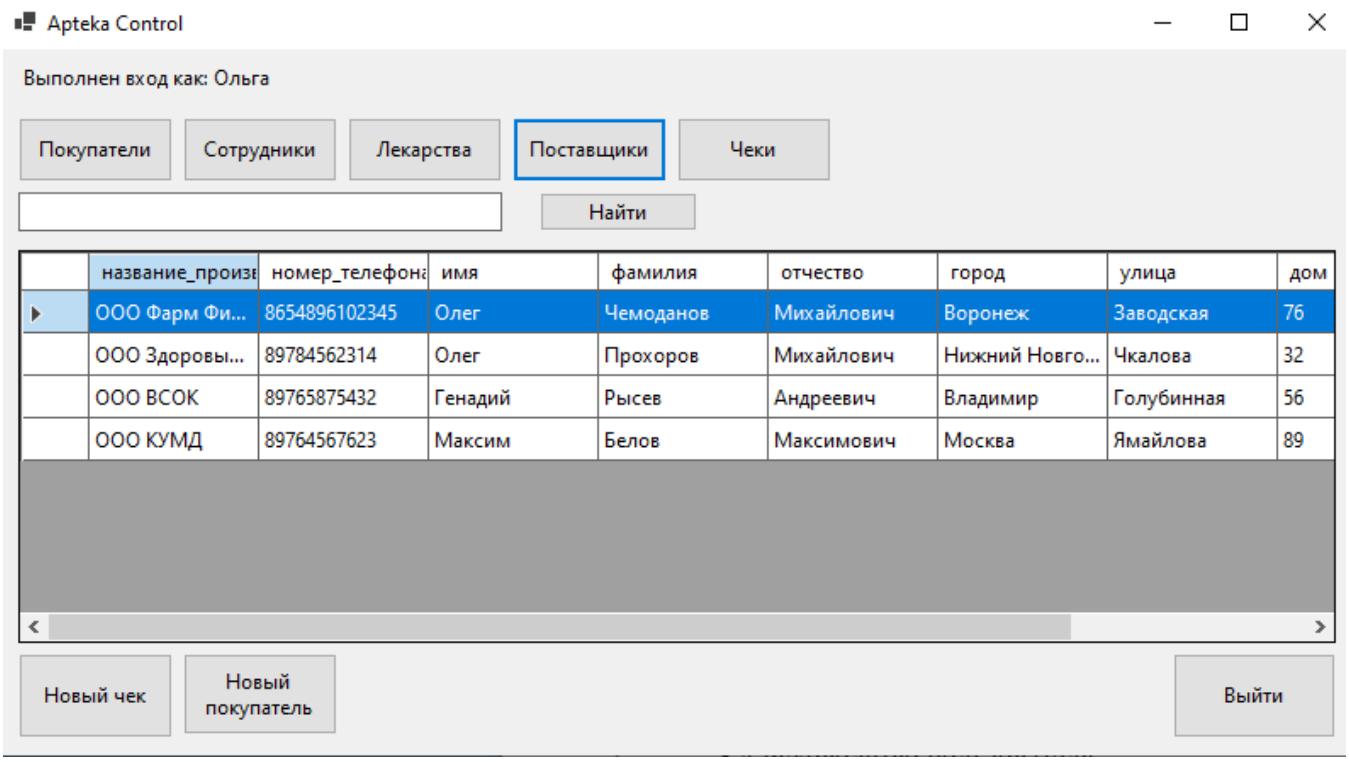


Рисунок 20: таблица «Сотрудники»



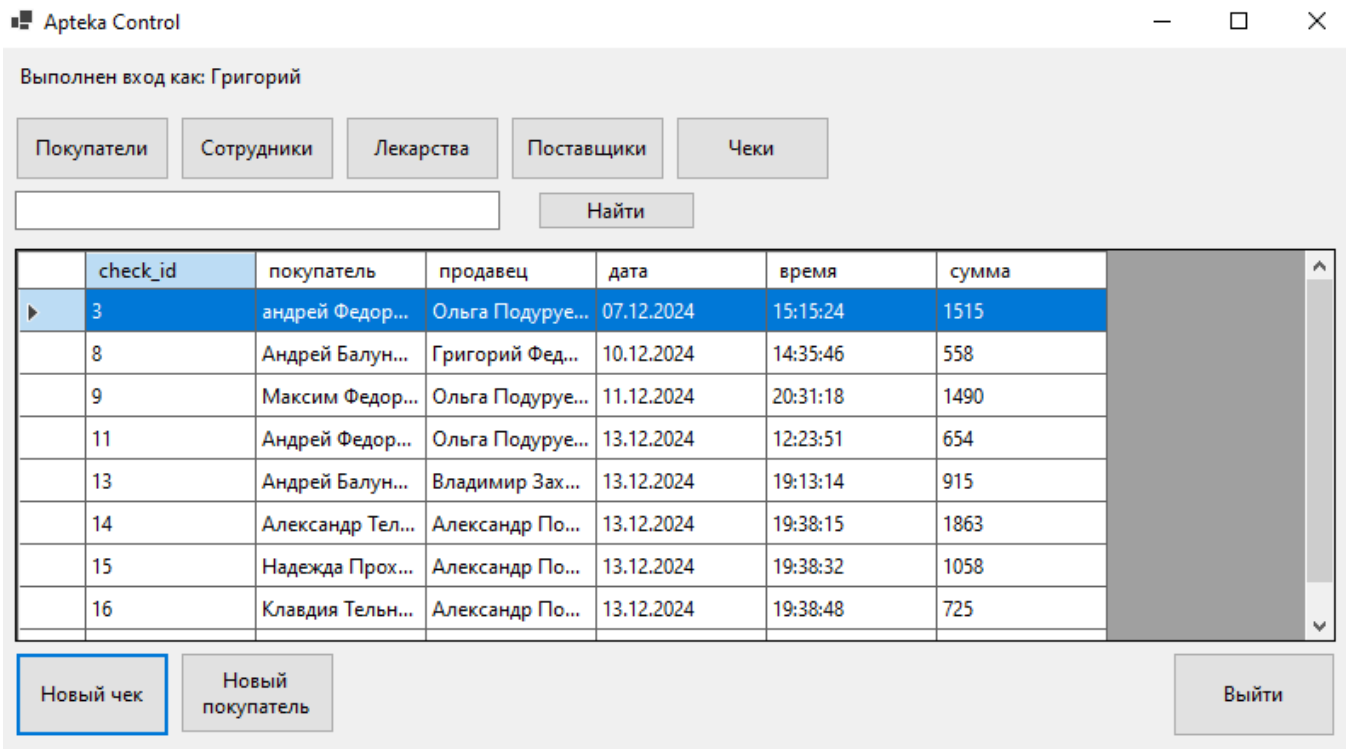
	название_произв	название_товара	тип	еденица_измере	количество_в_од	доступное_коли	цена	тр
▶	ООО Фарм Фи...	Нурофен	Таблетки	шт	20	66	149	да
	ООО Фарм Фи...	Глицин	Таблетки	шт	30	18	178	не
	ООО Здоровы...	Пектусин	Таблетки	шт	20	78	130	не
	ООО КУМД	Гексорал	Спрей	мл	200	62	502	не
	ООО ВСОК	Комбилипен	Таблетки	шт	20	133	173	да
	ООО Здоровы...	Амброксол(сир)	Сироп	мл	100	62	120	не
	ООО Здоровы...	Амброксол(таб)	Таблетки	шт	12	89	146	не
	ООО КУМД	Граммидин	Таблетки	шт	18	154	475	да

Рисунок 21: таблица «Лекарства»



	название_произв	номер_телефона	имя	фамилия	отчество	город	улица	дом
▶	ООО Фарм Фи...	8654896102345	Олег	Чемоданов	Михайлович	Воронеж	Заводская	76
	ООО Здоровы...	89784562314	Олег	Прохоров	Михайлович	Нижний Новго...	Чкалова	32
	ООО ВСОК	89765875432	Генадий	Рысев	Андреевич	Владимир	Голубинная	56
	ООО КУМД	89764567623	Максим	Белов	Максимович	Москва	Ямайлова	89

Рисунок 22: таблица «Поставщики»



check_id	покупатель	продавец	дата	время	сумма
3	андрей Федор...	Ольга Подуруе...	07.12.2024	15:15:24	1515
8	Андрей Балун...	Григорий Фед...	10.12.2024	14:35:46	558
9	Максим Федор...	Ольга Подуруе...	11.12.2024	20:31:18	1490
11	Андрей Федор...	Ольга Подуруе...	13.12.2024	12:23:51	654
13	Андрей Балун...	Владимир Зах...	13.12.2024	19:13:14	915
14	Александр Тел...	Александр По...	13.12.2024	19:38:15	1863
15	Надежда Прох...	Александр По...	13.12.2024	19:38:32	1058
16	Клавдия Тельн...	Александр По...	13.12.2024	19:38:48	725

Рисунок 23: таблица «Чеки»

3.4 Руководство пользователя

Для просмотра данных в таблицах необходимо нажать на кнопку нужной таблицы в главном окне приложения. Для поиска конкретной записи в таблице необходимо нажать на кнопку нужной таблицы, ввести в поле ввода ключего слово и нажать кнопку «Найти». Для оформления новой покупки необходимо нажать кнопу «Новый чек». В случае, если покупку делает новый клиент, необходимо зарегестрирвать его через кнопку «Новый покупатель» и занести необходимую информацию в форму. Для добавления лекарств в чек необходимо выбрать лекарство и его количество из выпадающего списка «товар», после чего нажать кнопку «Добавить в чек», в случае, если для продажи лекарства требуется рецепт, программа уведомит об этом сотрудника.

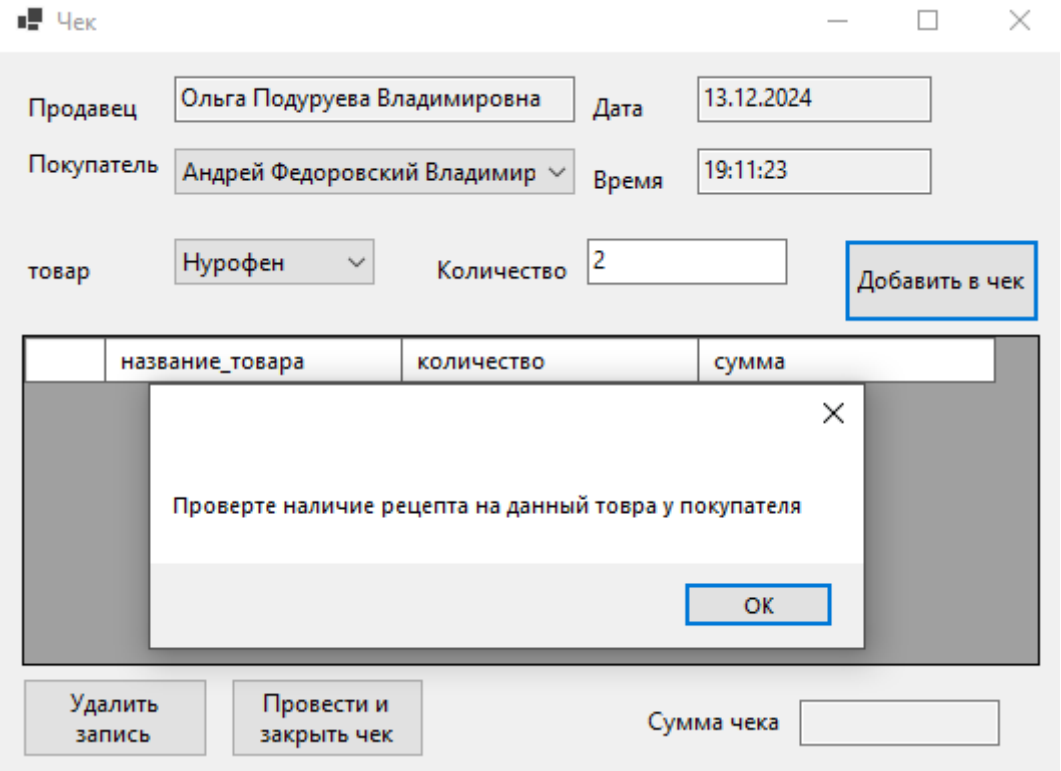


Рисунок 24: Напоминание продавцу при добавлении товара, требующего рецепта

После добавления лекарства в таблицу-корзину, программа автоматически выдит стоимость как отдельных элементов в чеке, так и всего чека в целом. В случае ошибочного заполнения чека, можно выбарть строку ошибочно добавленного лекарства в таблице и нажать кнопу «удалить запись» для удаления ошибочно добавленного лекарства. После завершения покупки необходимо нажать кнопу «Провести и закрыть чек» для закрытия чека, в случае, если на складе недостаточно товара для покупки, то программа уведомит об этом сотрудника и не закроет чек:

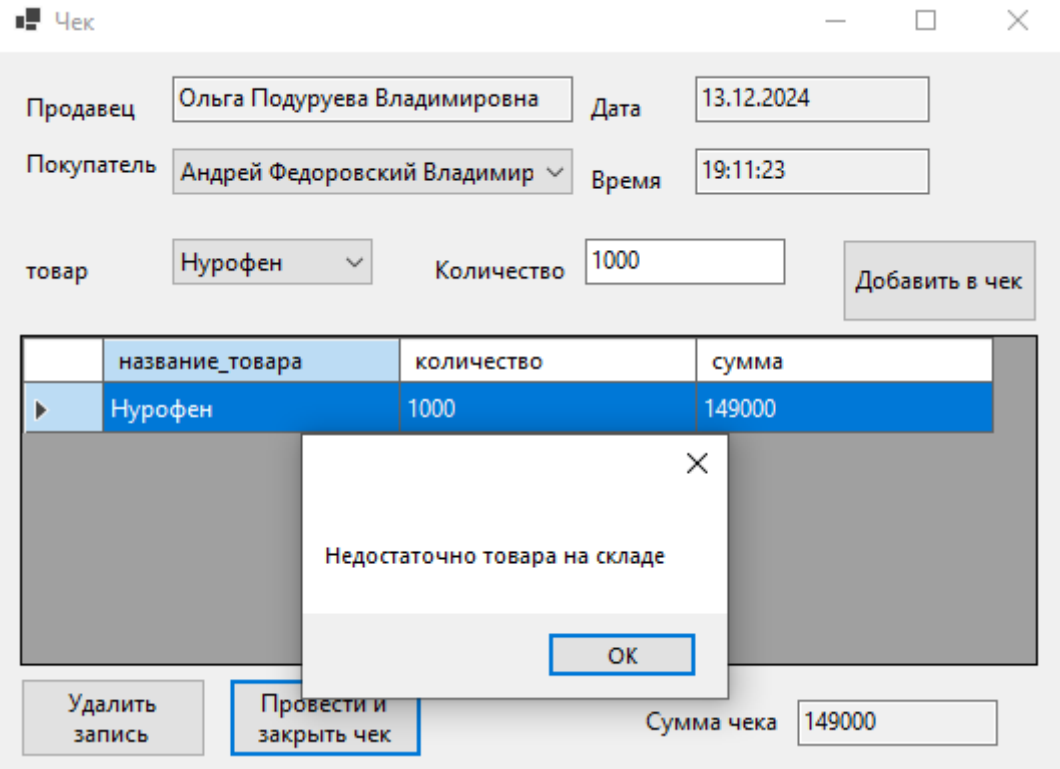


Рисунок 25: Сообщение о недостатке товара для совершения покупки.

В случае удачного закрытия чека программа так же уведомит пользователя:

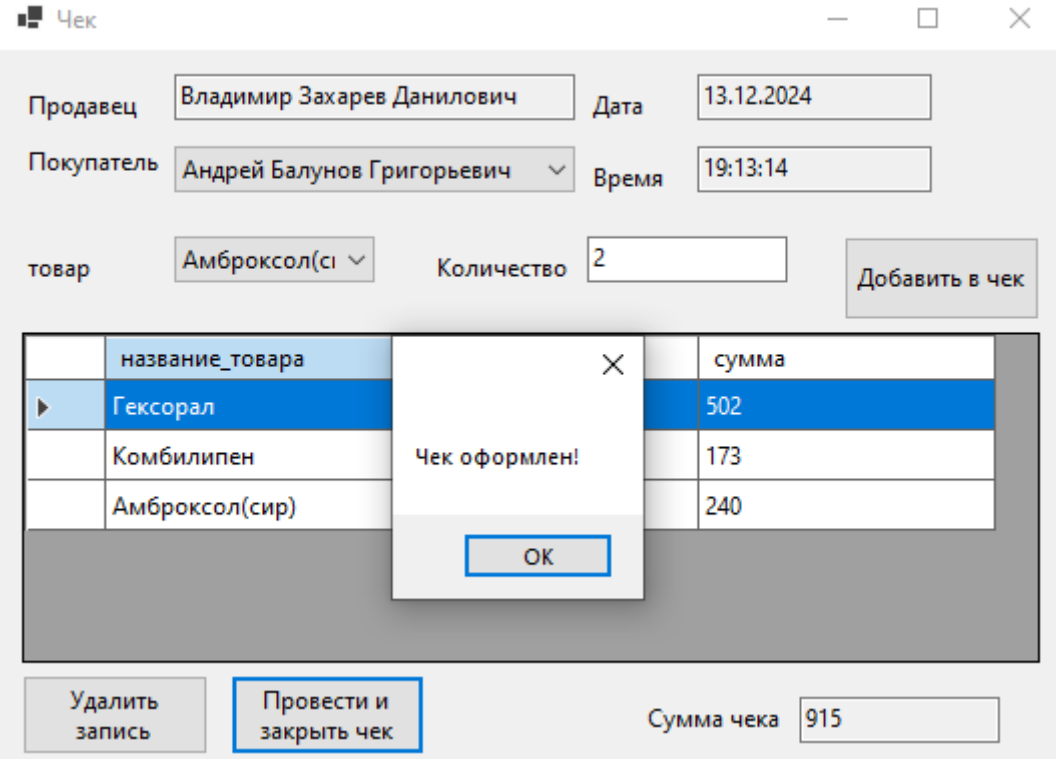


Рисунок 26: Закрытие чека

3.5 Руководство администратора

Для корректной работы на ПК должен быть установлен пакет .NET 8.0, а файл базы данных должен находится в корневом каталоге программы.

Администратор может просматривать, редактировать и удалять любые записи в базе данных кроме записей с чеками, их можно только просматривать:

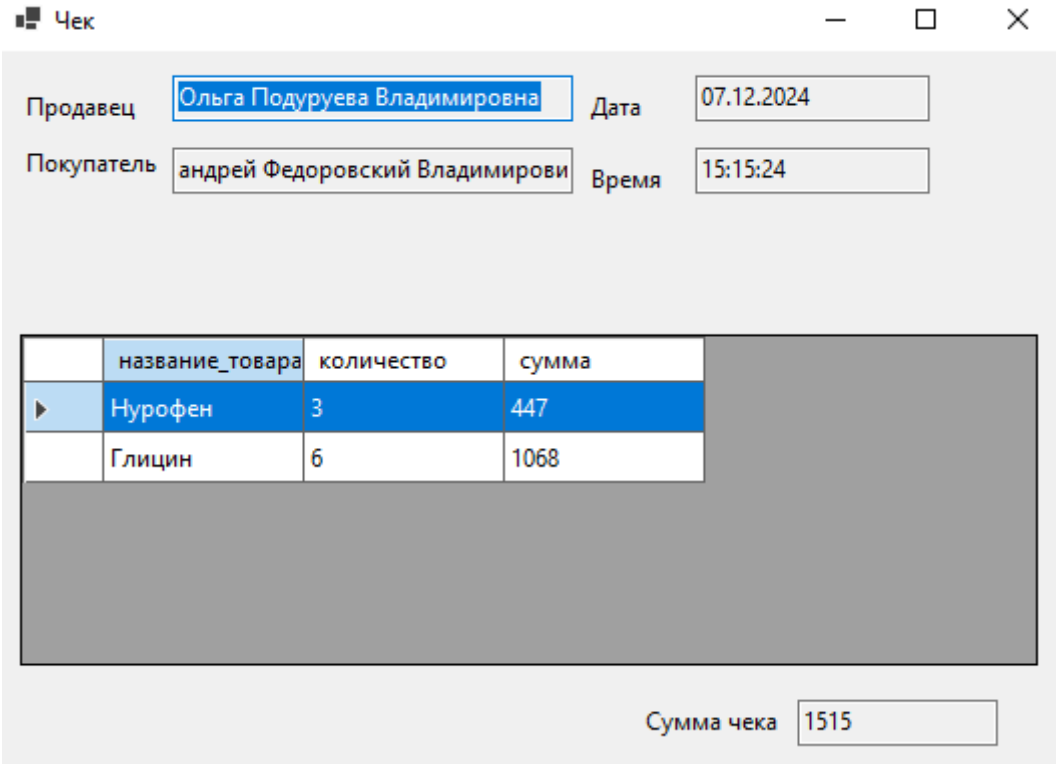


Рисунок 27: Пример просмотра чека

Добавление новый записей в базе данных происходит через элемент меню «Добавить запись», где необходимо выбрать таблицу для добавления записи туда и заполнить соответсвтующую форму, для редактирования или удаления записи, необходимо в главной форме выбрать нужную запись в таблице и нажать кнопку «Изменить» или «Удалить» соответственно. Для обучения персонала эксплуатации программы, необходимо и достаточно предоставить к прочтению руководство пользователя из пункта 3.4

3.6 Подробное описание программы

Структура программы:

- Form1.cs – Главная форма администратора
- Form15.cs – Главная форма сотрудника
- FormCustomerADD.cs – Форма для добавления и изменения записей о покупателях
- FormEmployeeADD.cs – Форма для добавлния и изменения записей о сотрудниках
- FormProviderADD.cs – Форма для добавления и изменения записей о поставщиках
- FormMedicineADD.cs – Форма для добавления и изменения записей о лекарствах
- UltimateCheckForm.cs – Форма для добавления записей о чеках и подробном просмотре чеков
- FormRegistration.cs – Форма авторизации

3.6.1 Описание функций в форме Form1:

- FillDataGridView – используется как дополнение для вывода select запросов для заполнения пустых полей.
- SeeCustomers(и другие название таблиц) – используется для полненения select запросов для таблиц и вывода результата на экран
- CheckAvailableTables – используется для проверки подключения к базе данных и вывода на экран списка доступных таблиц
- CheckDatabaseFile – используется для проверки подлючения к базе данных
- Customer_button_Click(И другие название таблиц – вызывает метод функцию «See{название_таблицы}» и руководит доступом к кнопкам «удалить» и «изменить»
- delete_button_Click – удаляет выбранную запись из таблицы
- button_search_Click – Произовдит select запрос с ключевым словом для выбранной таблицы

- change_button_Click – вызывает форму для изменения выбранной записи в выбранной таблице
- покупателитoolStripMenuItem1_Click(и другие названия таблиц) – вызывает форму для добавления записи в выбранную таблицу
- Explore_check_button_Click – вызывает форму для просмотра подробной информации о чеке
- dataGridView1_DataError – используется для обработок ошибок с таблицами

3.6.2 Описание функций в форме Form15

- Большинство функций повторяют форму Form1 за исключением отсутствия функций для удаления, добавления и изменения записей во все таблицы, кроме покупателей и чеков
- New_check_button_Click – вызывает форму для добавления нового чека
- new_customer_button_Click – вызывает форму для добавления нового покупателя

3.6.3 Описание функций в формах Form{название_таблицы}ADD

- button1_Click – в зависимости от подачи id записи в форму, данная кнопка или добавляет запись в таблицы, или изменяет уже имеющуюся
- Form{имя}_Load – очищает строки с сообщениями об ошибке при загрузке формы
- textBox{номер}_KeyPress – Запрещает писать в текстбокс запрещённые символы

3.6.4 Описание функций в форме UltimateCheckForm

- seePurchases – используется для подробного просмотра чека
- button_add_tovar – используется для добавления лекарства в чек
- comboBox{номер}_SelectedIndexChanged – используется для получения ифнормации из комбобокосов
- button_delete_tovar – используется для удаления ошибочно добавленного товара в чек
- Add_check_button – используется для проведения и закрытия чека

4. Тестирование АИС

При завершении разработки программы необходимо провести проверку на её работоспособность при различных действиях пользователей. Программа должна корректно отображать и выполнять действия пользователя в любом случае. В исключительных моментах должны выводиться окна с сообщениями, поясняющие ситуацию.

Проверим, как программа будет реагировать на некоторые действия пользователя.

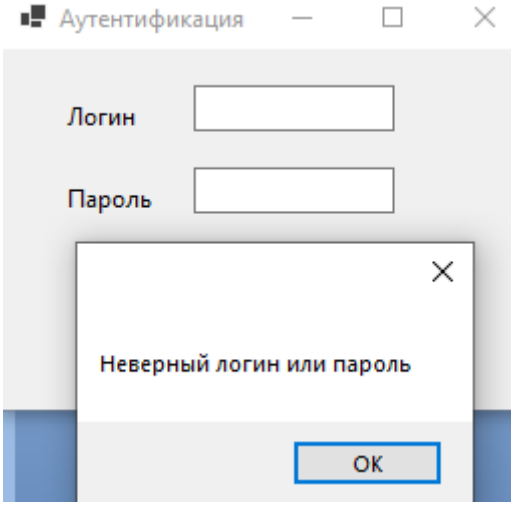


Рисунок 28: Проваленная аутентификация

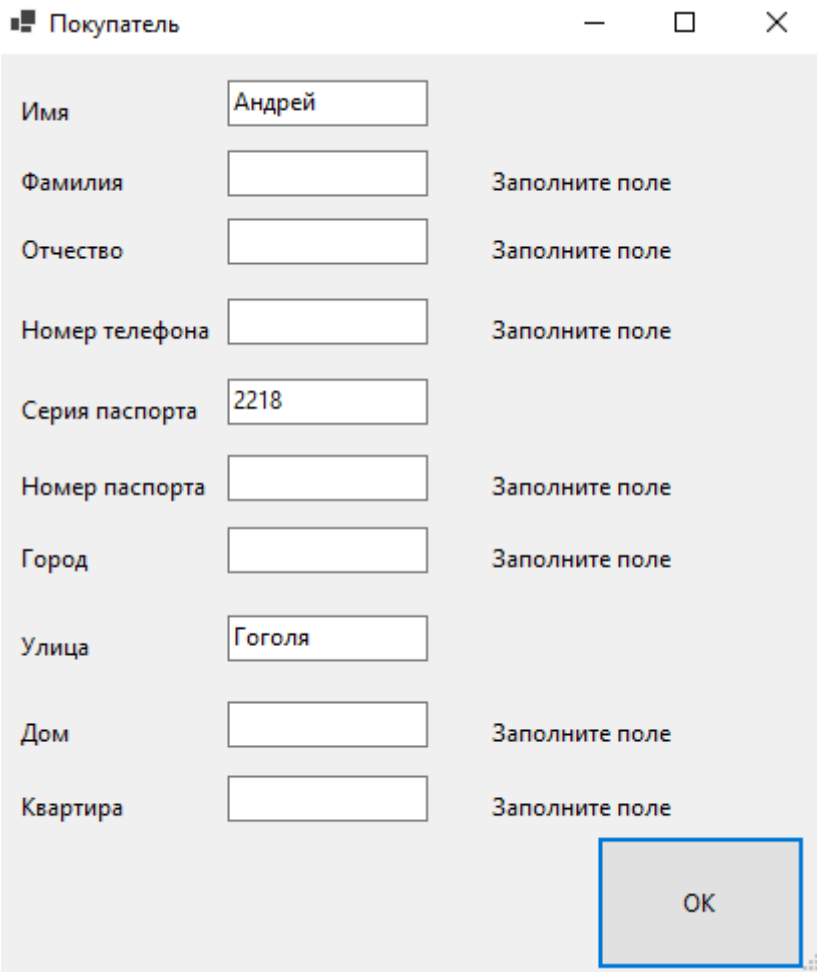


Рисунок 29: Реакция на пустые поля при добавлении и изменении записей

Следует отметить, что явные ошибки, такие как заполнение числовых полей буквами, исключены ввиду запрета ввода запрещённых символов в соответствующие поля.

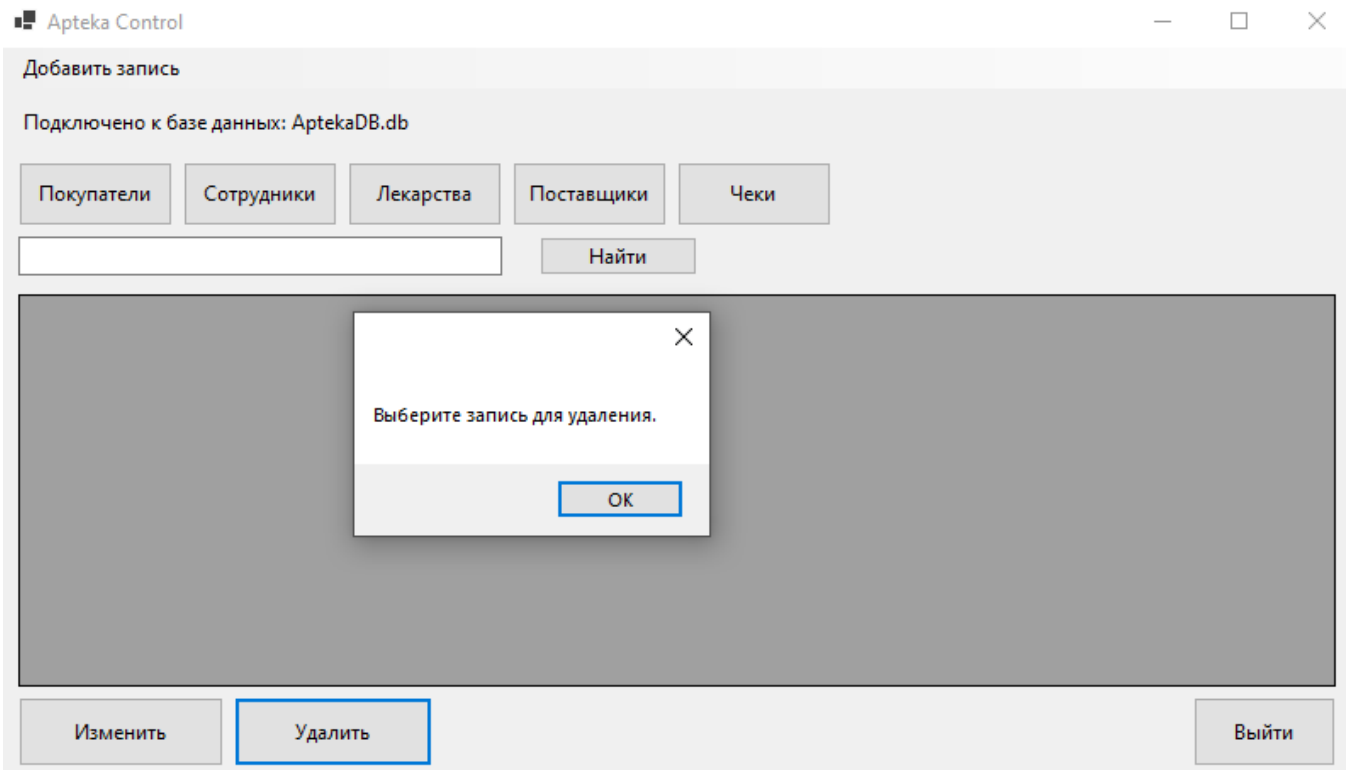


Рисунок 30: Реакция на удаление ничего

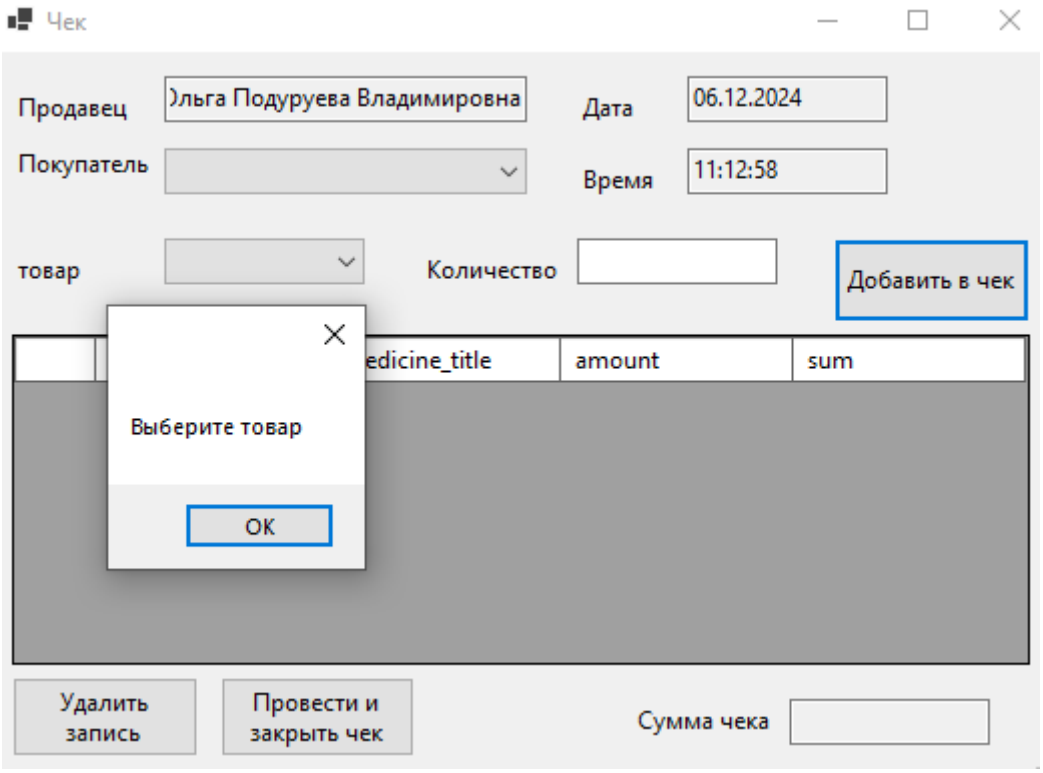


Рисунок 31: Ошибка при заполнении чека

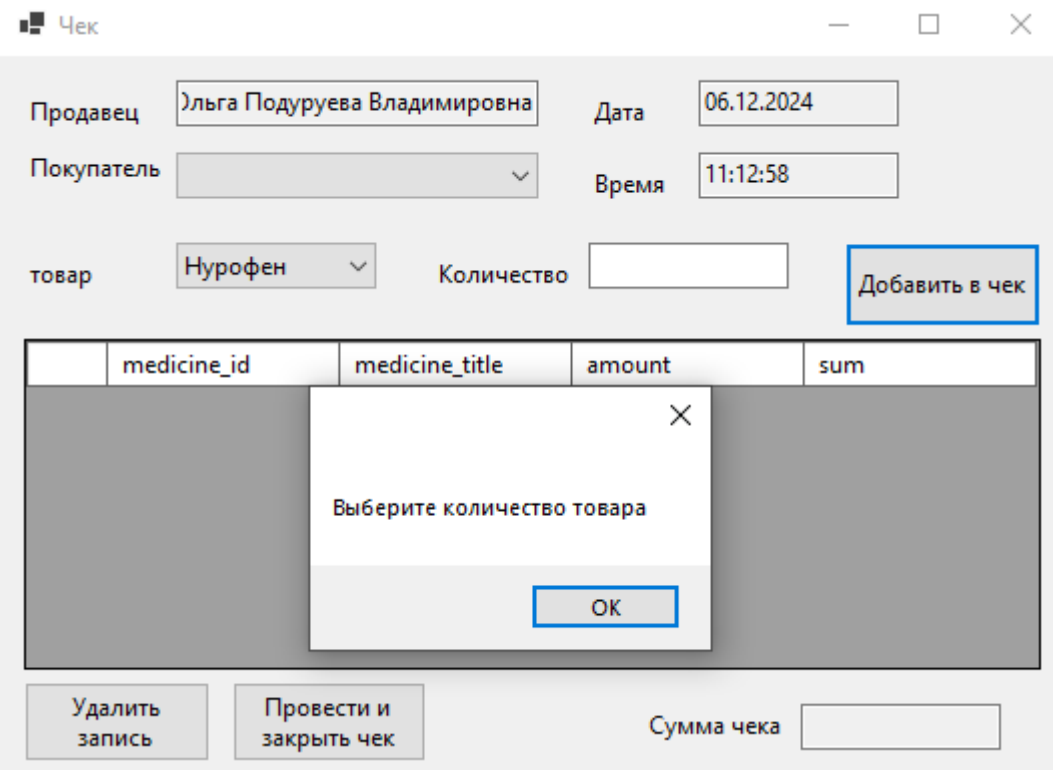


Рисунок 32: Ошибка при заполнении чека

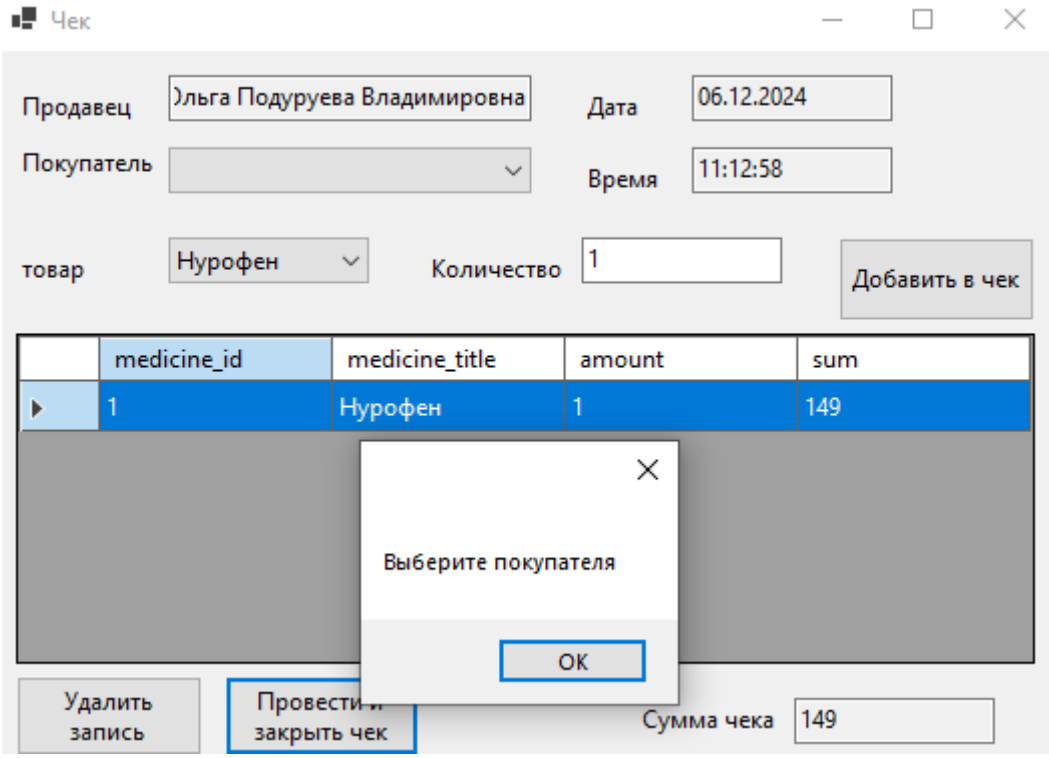


Рисунок 33: Ошибка при заполнении чека

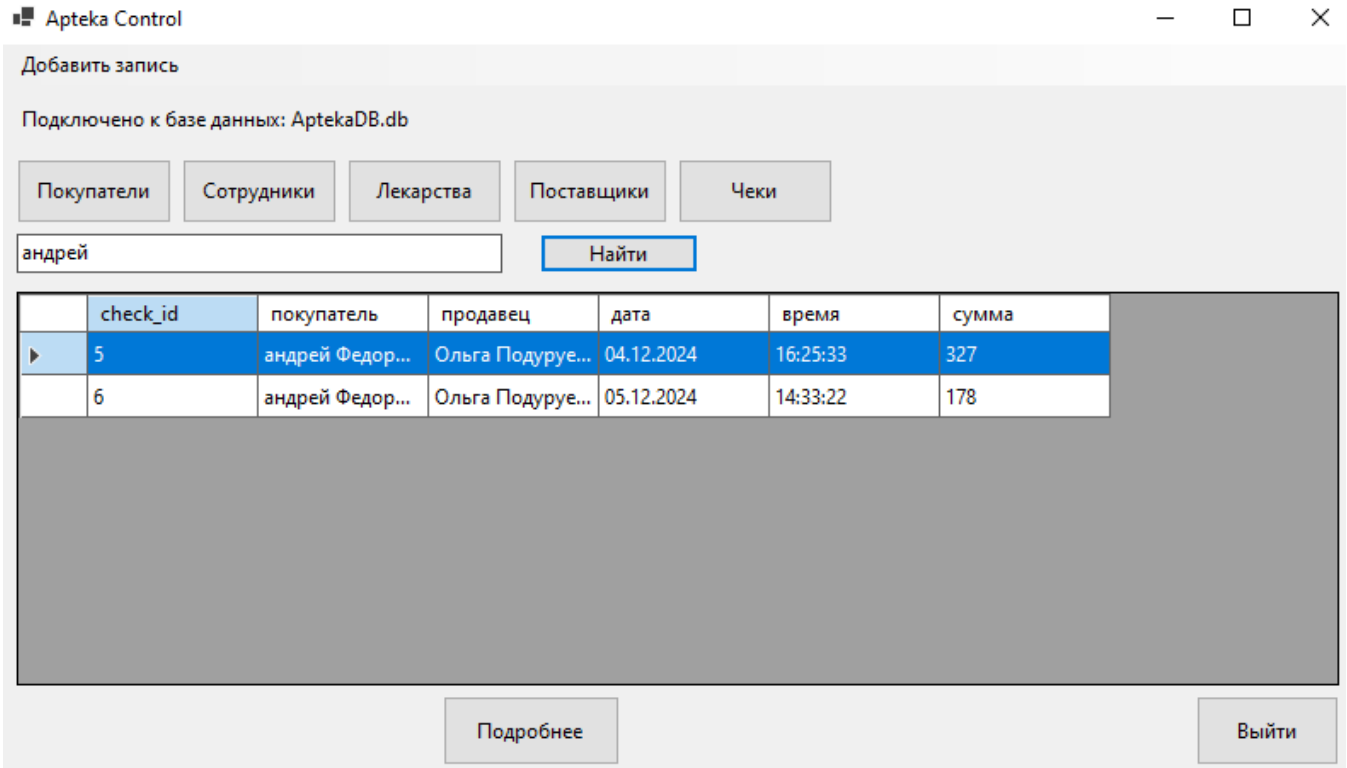


Рисунок 34: Пример работы поиска по ключевому слову

Заключение

В процессе выполнения данной курсовой работы была успешно разработана программа, использующая среду Visual Studio и язык программирования C#. Взаимодействие с базой данных было реализовано с использованием библиотеки System.Data.SQLite, обеспечивая эффективное хранение и управление данными.

В процессе программирования активно использовалась сторонняя литература, в основном, представленная в виде веб-ресурсов и электронных книг, что содействовало успешному решению возникающих задач.

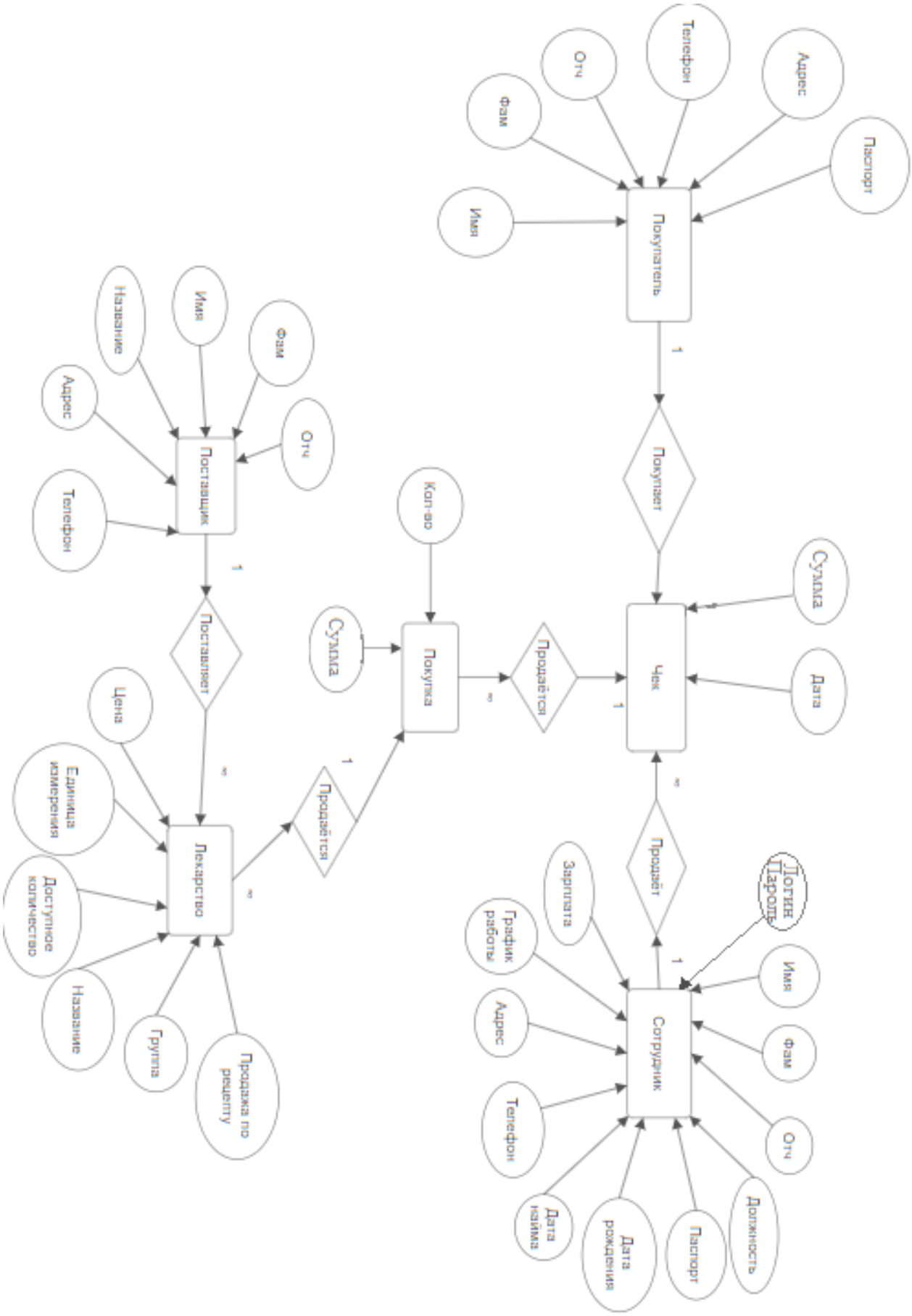
Разработанная программа отвечает всем поставленным требованиям, обеспечивая удовлетворительное взаимодействие с базой данных без критических ошибок.

Список литературы

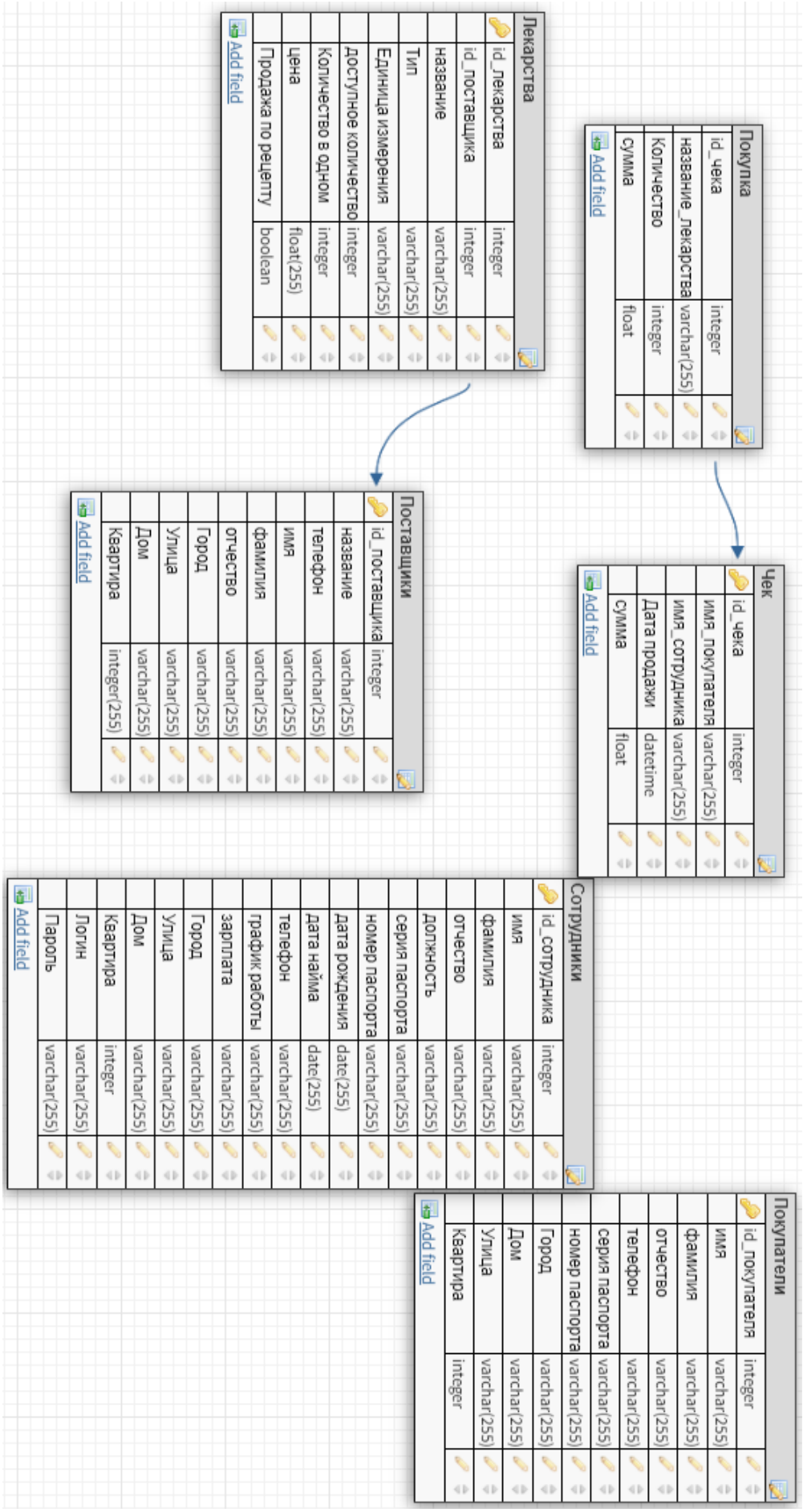
1. Карпова И.П. Базы данных. Учебное пособие. – СПб.: Питер, 2013 г. – 240 с.
2. Лапшин, Ю. Л. "Проектирование информационных систем. Методические указания." Издательский центр "Академия", 2017 г. – 26 с.
3. Лисин, С. Ю. "Разработка клиент-серверных приложений в Microsoft SQL Server." ДМК Пресс, 2009 г. – 61 с.
4. Тарасов, С. В. СУБД для программиста. Базы данных изнутри / С. В. Тарасов. — Москва : СОЛОН-Пресс, 2018. — 320 с. — ISBN 978-2-7466-7383-0. — Текст : электронный // Цифровой образовательный ресурс IPR SMART : [сайт]. — URL: https://www.iprbookshop.ru/90409.html (дата обращения: 11.12.2024). — Режим доступа: для авторизир. пользователей

Приложение 1. Модели данных

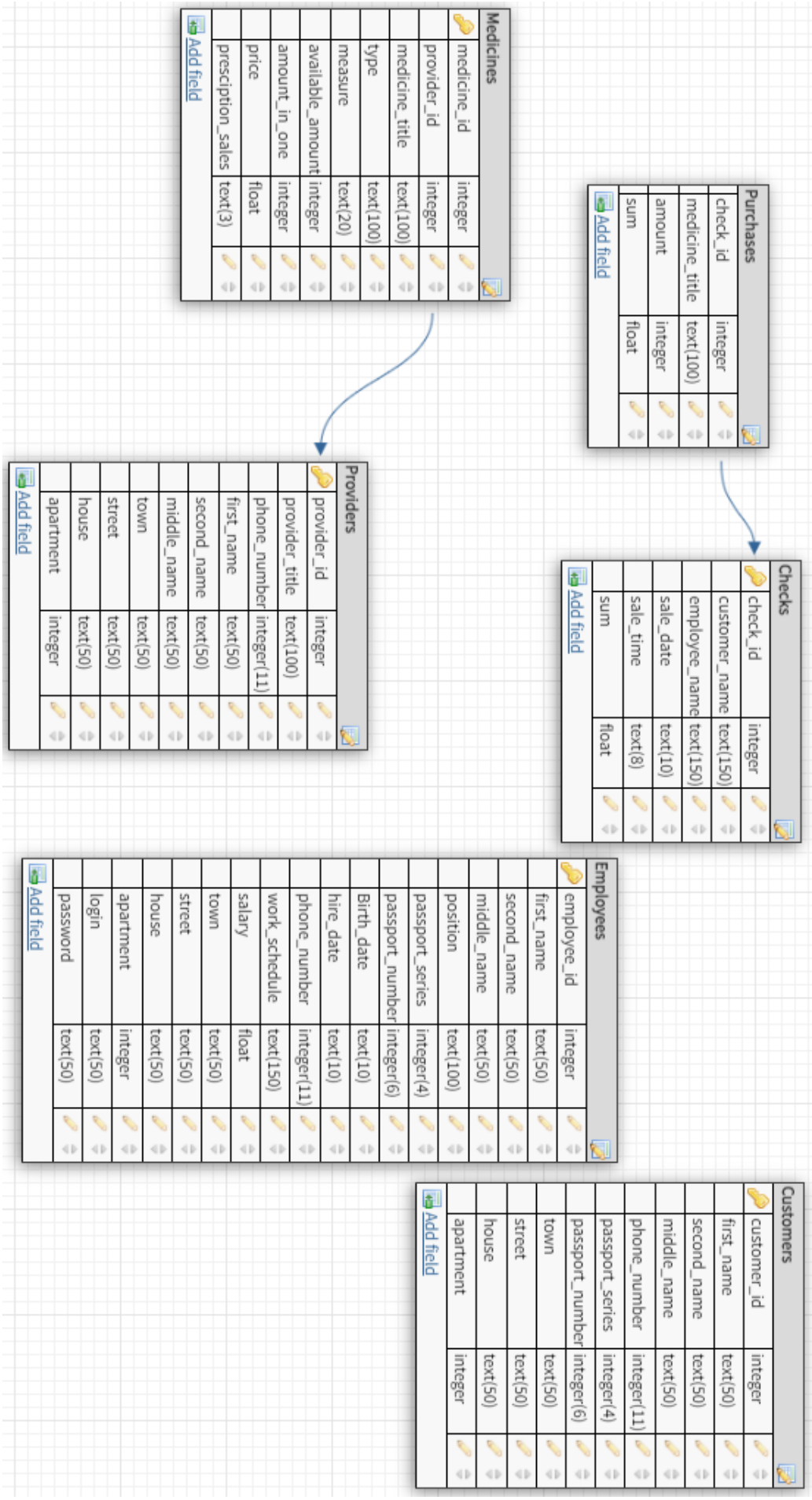
Концептуальная модель



Логическая модель



Физическая модель

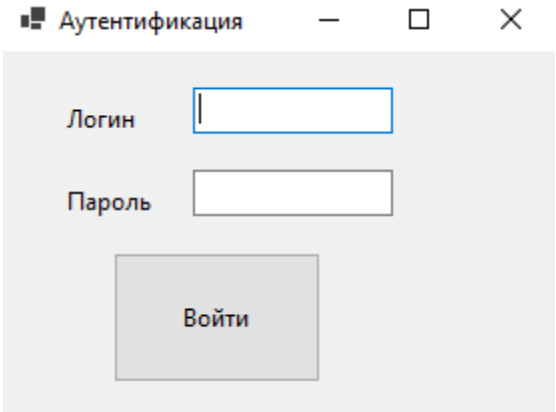


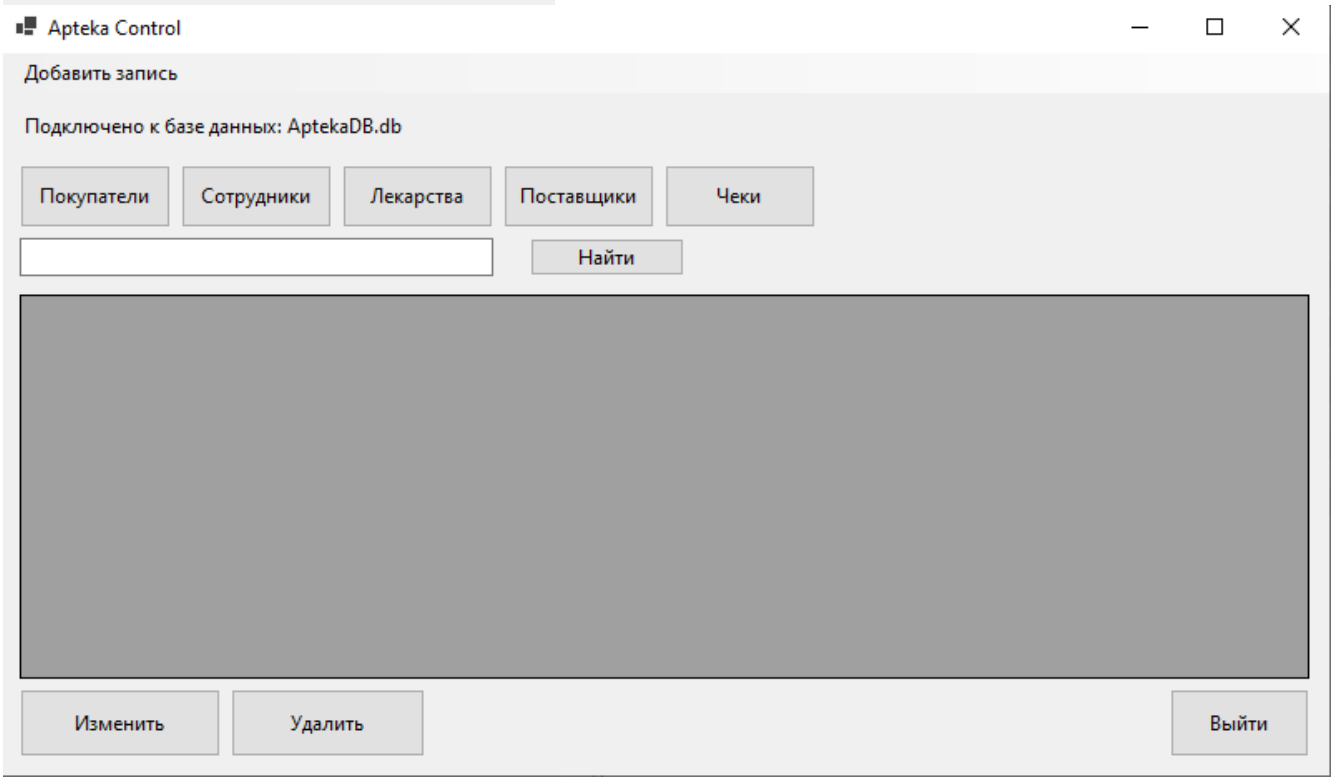
Приложение 2. Текст программы

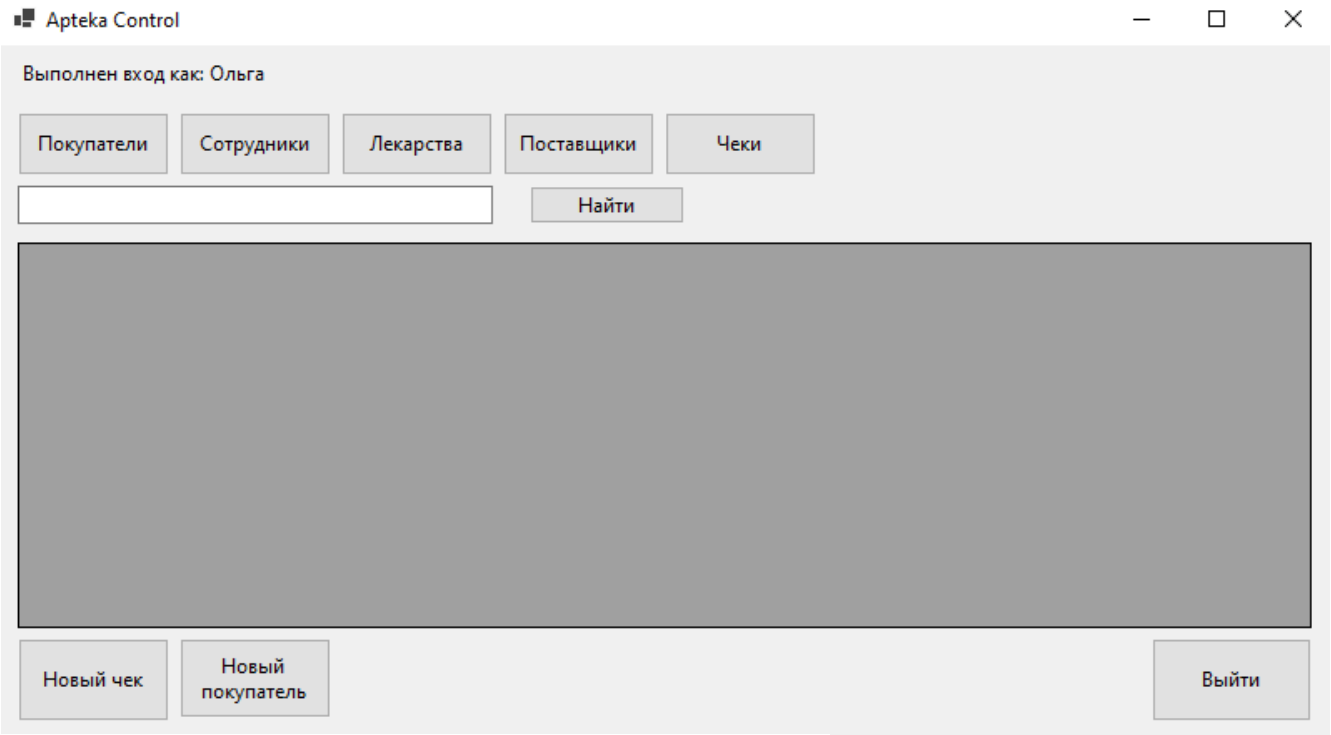
Ссылка на весь проект:

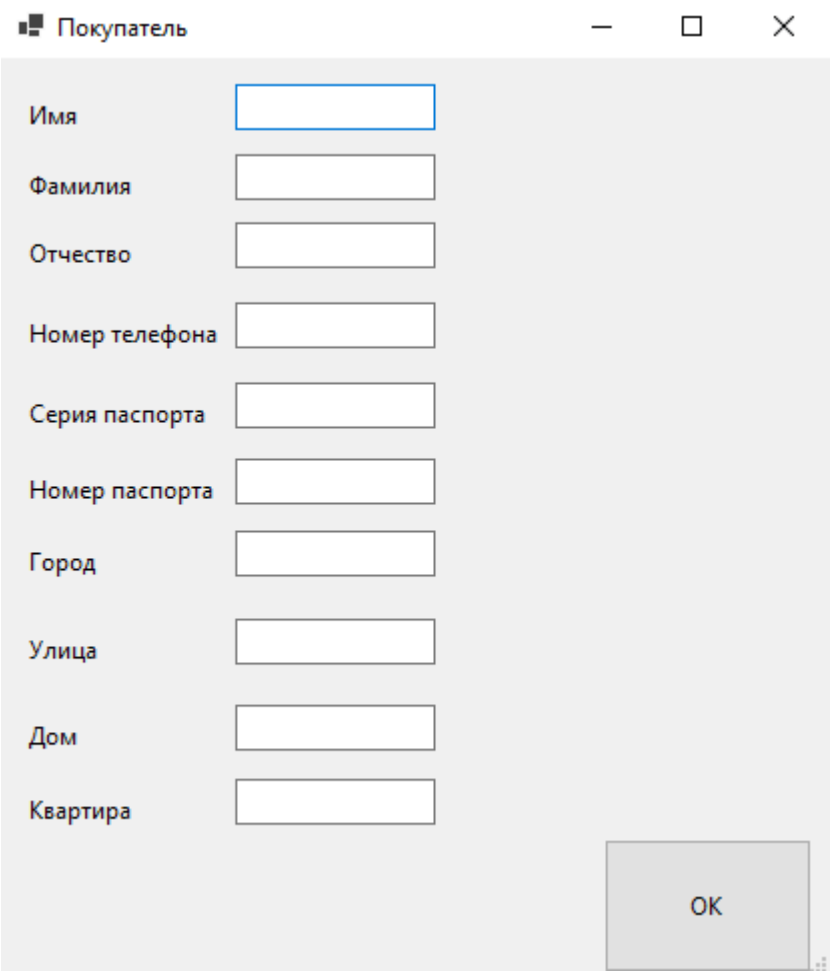
https://github.com/Pacmanoidum/Apteka_control

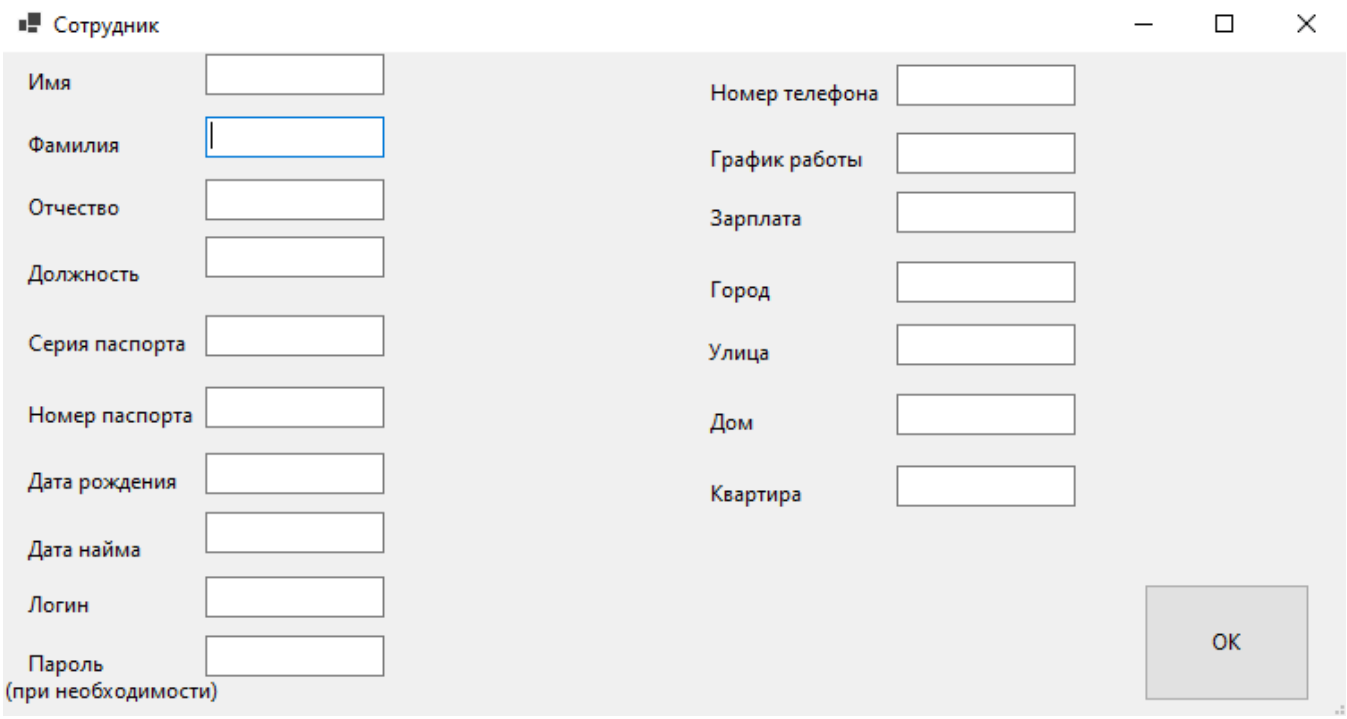
Приложение 3: Снимки окон программы

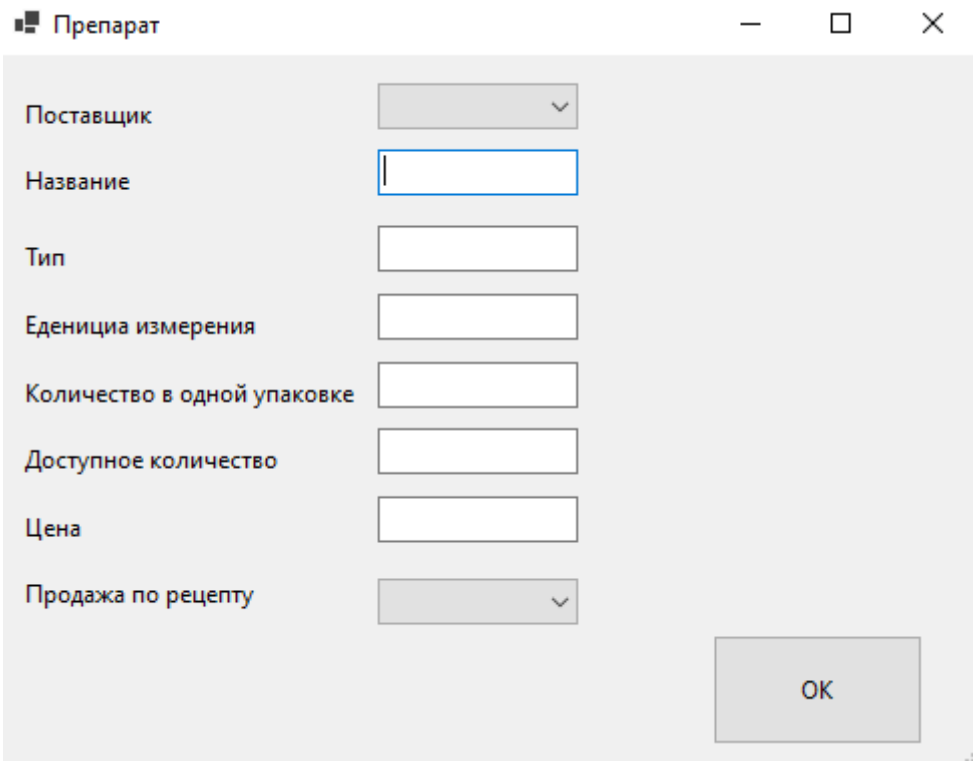


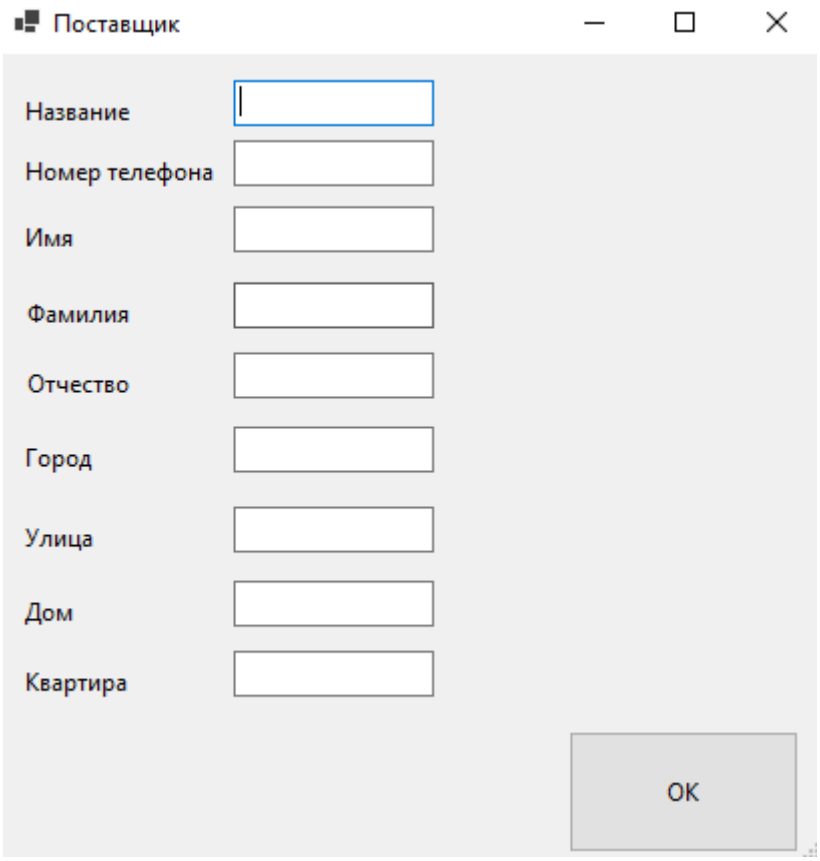


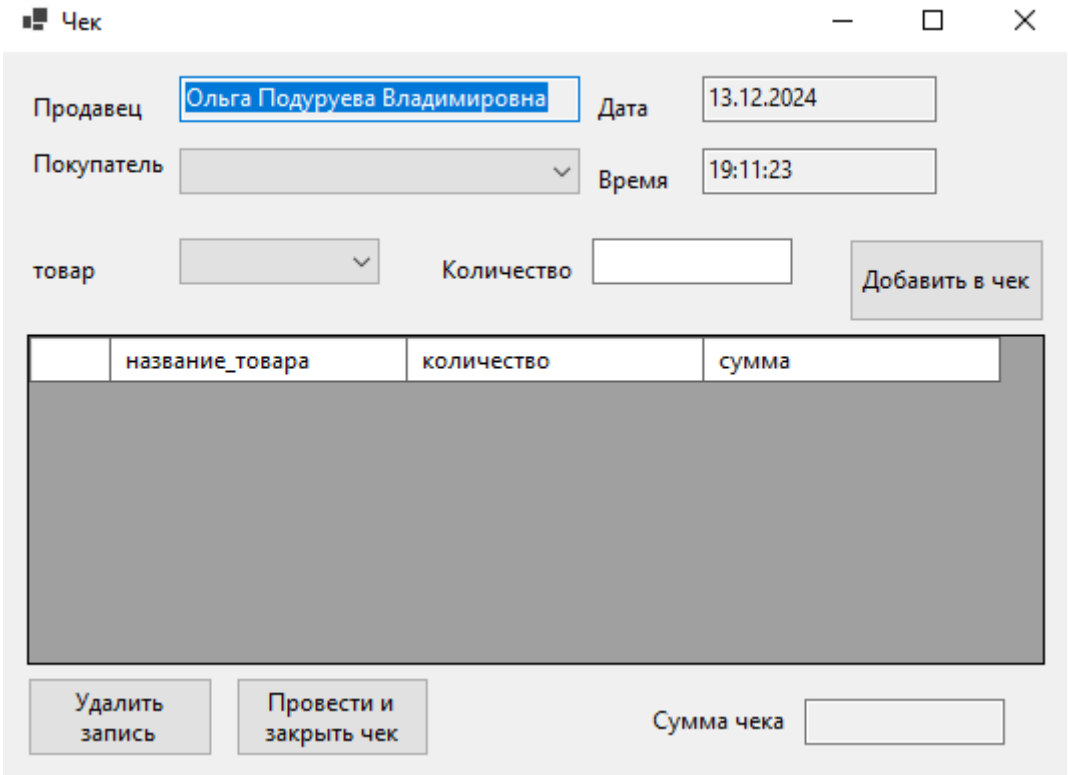

Apteka Control
Выполнен вход как: Ольга
Покупатели
Сотрудники
Лекарства
Поставщики
Чеки
Найти
Новый чек
Новый покупатель
Выйти

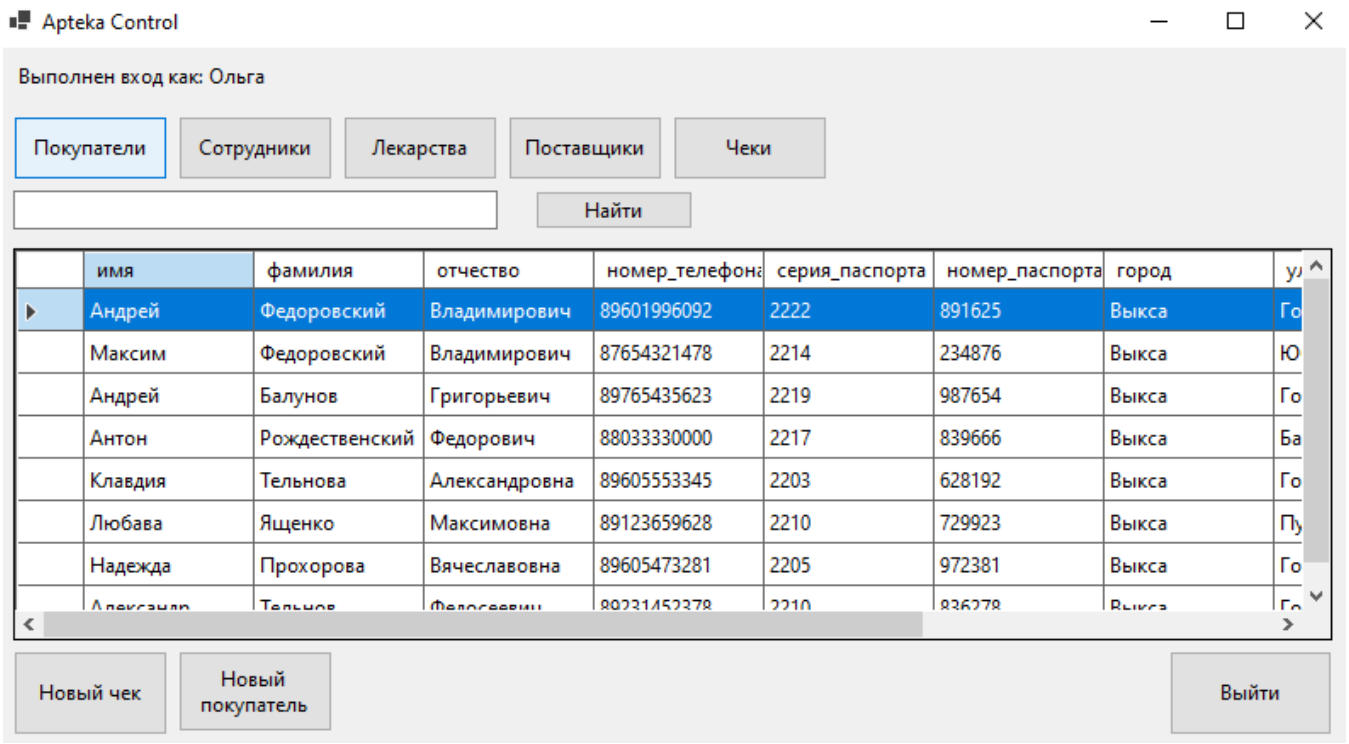

Покупатель
Имя
Фамилия
Отчество
Номер телефона
Серия паспорта
Номер паспорта
Город
Улица
Дом
Квартира
OK


Сотрудник
Имя
Фамилия
Отчество
Должность
Серия паспорта
Номер паспорта
Дата рождения
Дата найма
Логин
Пароль
(при необходимости)
Номер телефона
График работы
Зарплата
Город
Улица
Дом
Квартира
OK

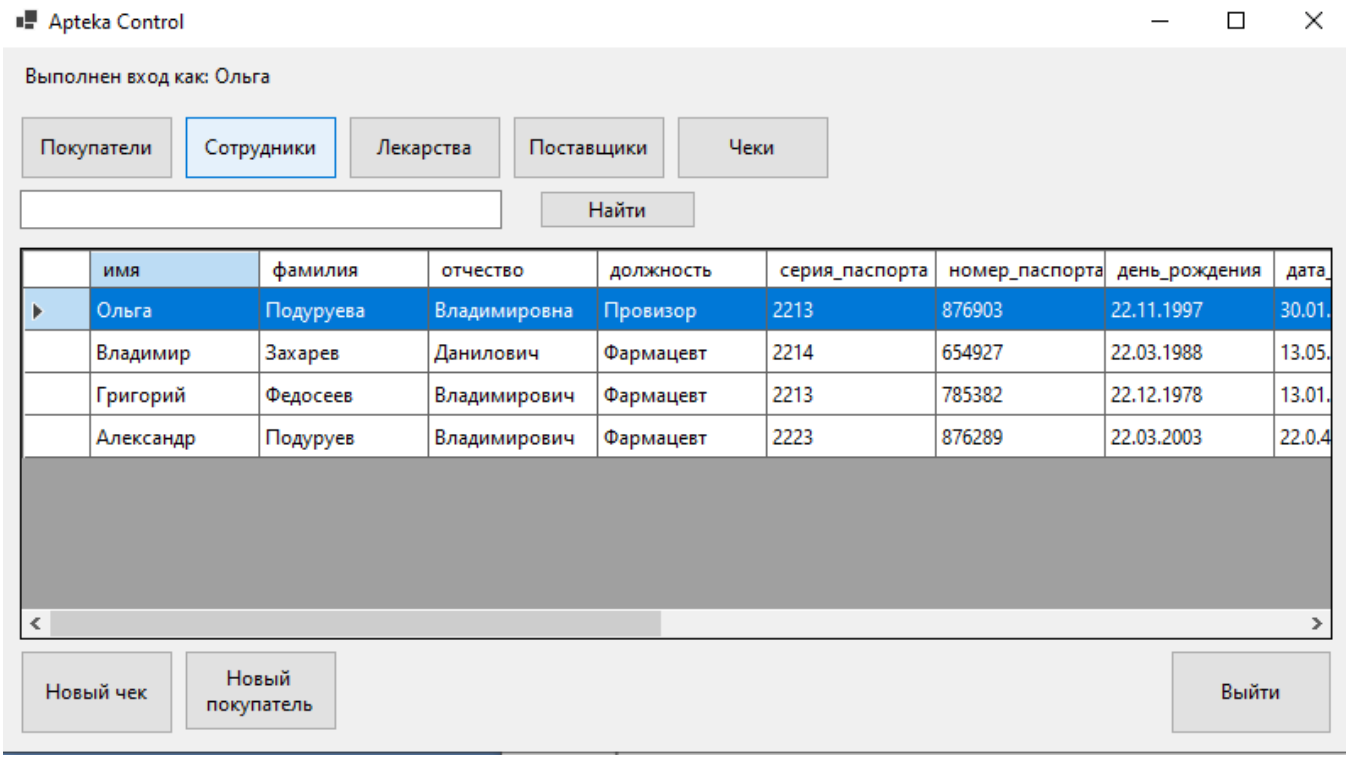

Препарат
Поставщик
Название
Тип
Еденицина измерения
Количество в одной упаковке
Доступное количество
Цена
Продажа по рецепту
OK


Поставщик
Название
Номер телефона
Имя
Фамилия
Отчество
Город
Улица
Дом
Квартира
OK

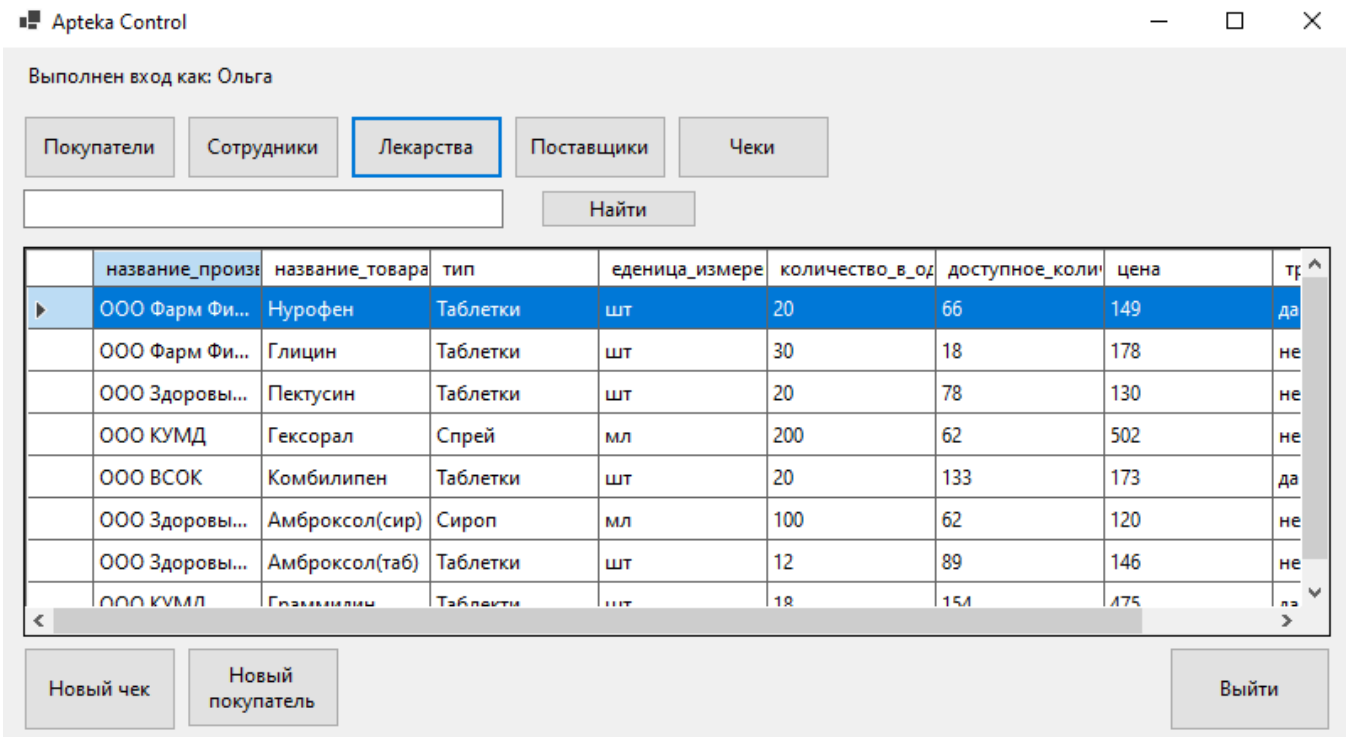

Чек
Продавец
Ольга Подуруева Владимировна
Дата
13.12.2024
Покупатель
Время
19:11:23
товар
Количество
Добавить в чек
название_товара
количество
сумма
Удалить запись
Провести и закрыть чек
Сумма чека



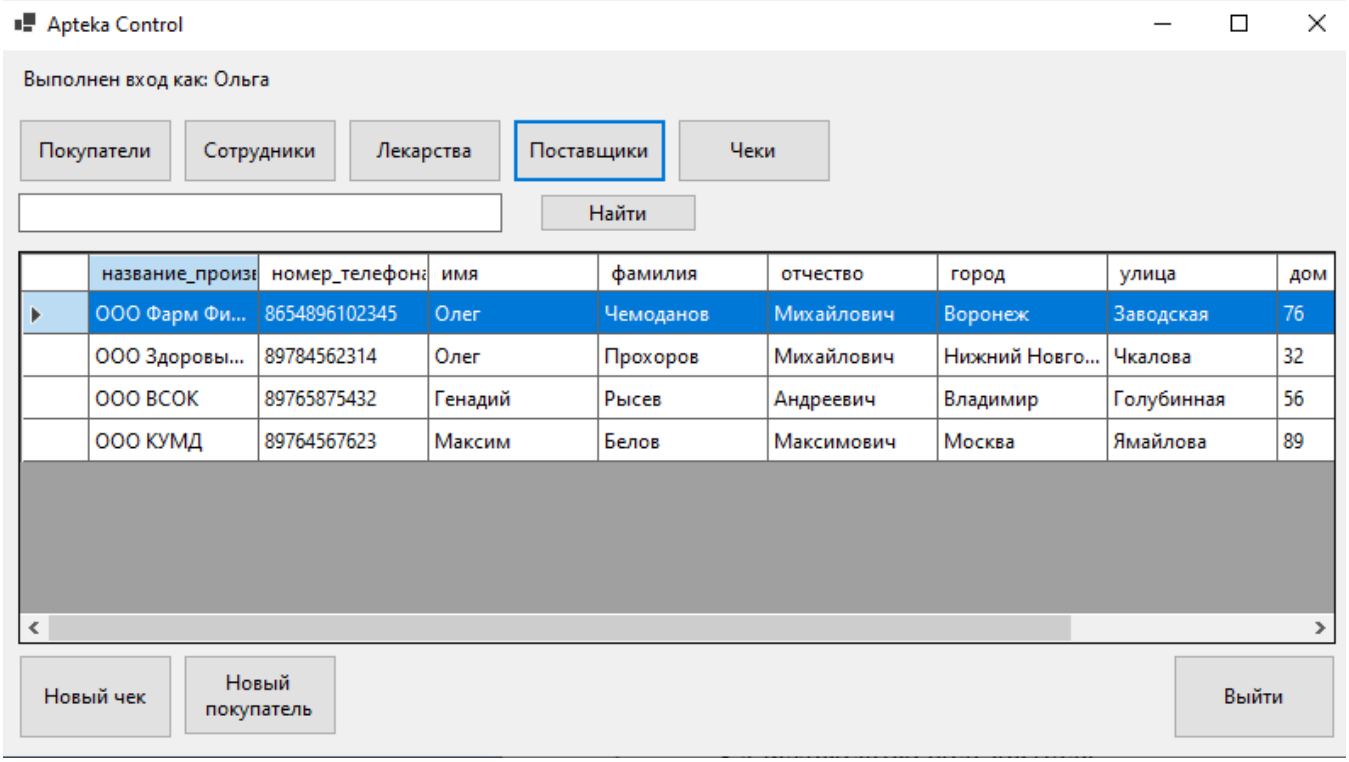
имя	фамилия	отчество	номер_телефона	серия_паспорта	номер_паспорта	город	ул
Андрей	Федоровский	Владимирович	89601996092	2222	891625	Выкса	Го
Максим	Федоровский	Владимирович	87654321478	2214	234876	Выкса	Ю
Андрей	Балунов	Григорьевич	89765435623	2219	987654	Выкса	Го
Антон	Рождественский	Федорович	88033330000	2217	839666	Выкса	Ба
Клавдия	Тельнова	Александровна	89605553345	2203	628192	Выкса	Го
Любава	Ященко	Максимовна	89123659628	2210	729923	Выкса	Пу
Надежда	Прохорова	Вячеславовна	89605473281	2205	972381	Выкса	Го
Александр	Тельнов	Федосеевич	89231452378	2210	836278	Выкса	Го



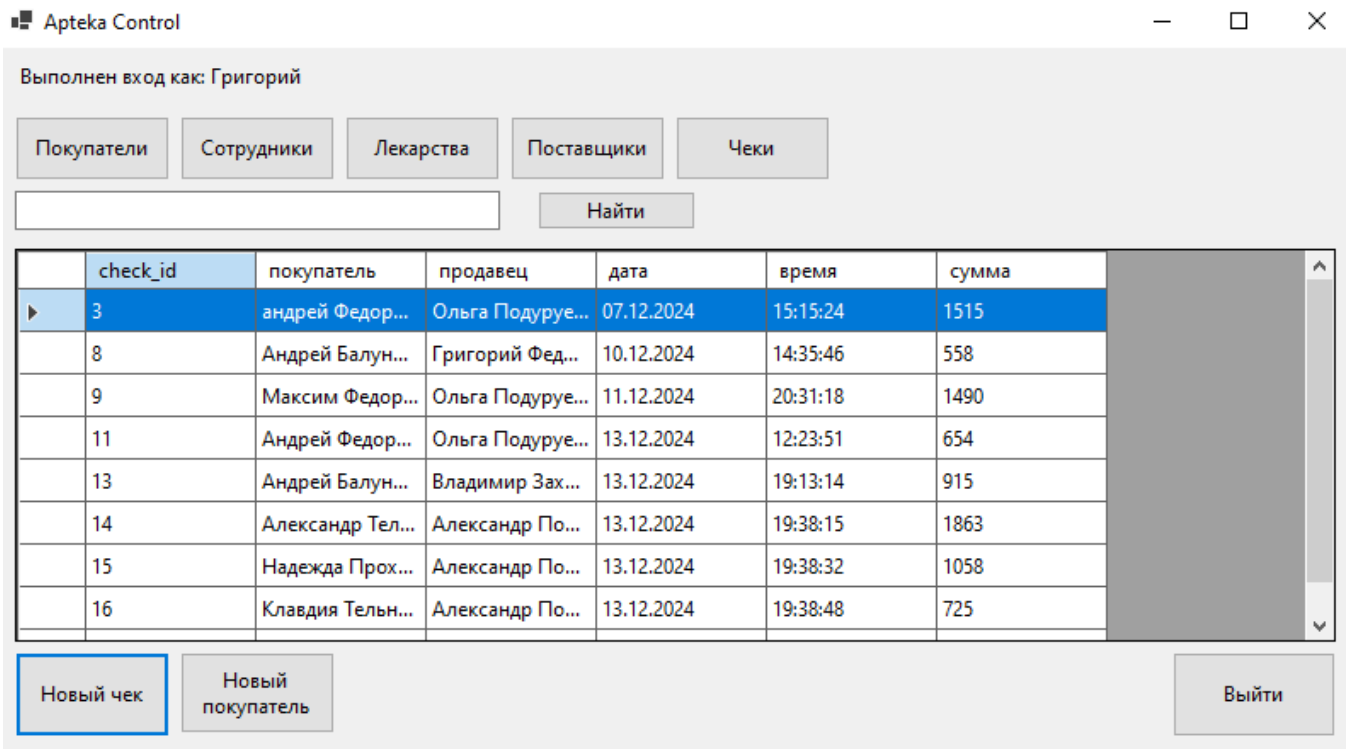
имя	фамилия	отчество	должность	серия_паспорта	номер_паспорта	день_рождения	дата_
Ольга	Подуруева	Владимировна	Провизор	2213	876903	22.11.1997	30.01.
Владимир	Захарев	Данилович	Фармацевт	2214	654927	22.03.1988	13.05.
Григорий	Федосеев	Владимирович	Фармацевт	2213	785382	22.12.1978	13.01.
Александр	Подуруев	Владимирович	Фармацевт	2223	876289	22.03.2003	22.0.4



	название_произв	название_товара	тип	еденица_измере	количество_в_од	доступное_коли	цена	тр
▶	ООО Фарм Фи...	Нурофен	Таблетки	шт	20	66	149	да
	ООО Фарм Фи...	Глицин	Таблетки	шт	30	18	178	не
	ООО Здоровы...	Пектусин	Таблетки	шт	20	78	130	не
	ООО КУМД	Гексорал	Спрей	мл	200	62	502	не
	ООО ВСОК	Комбилипен	Таблетки	шт	20	133	173	да
	ООО Здоровы...	Амброксол(сир)	Сироп	мл	100	62	120	не
	ООО Здоровы...	Амброксол(таб)	Таблетки	шт	12	89	146	не
	ООО КУМД	Граммидин	Таблетки	шт	18	154	475	да



	название_произв	номер_телефона	имя	фамилия	отчество	город	улица	дом
▶	ООО Фарм Фи...	8654896102345	Олег	Чемоданов	Михайлович	Воронеж	Заводская	76
	ООО Здоровы...	89784562314	Олег	Прохоров	Михайлович	Нижний Новго...	Чкалова	32
	ООО ВСОК	89765875432	Генадий	Рысев	Андреевич	Владимир	Голубинная	56
	ООО КУМД	89764567623	Максим	Белов	Максимович	Москва	Ямайлова	89



check_id	покупатель	продавец	дата	время	сумма
3	андрей Федор...	Ольга Подуруе...	07.12.2024	15:15:24	1515
8	Андрей Балун...	Григорий Фед...	10.12.2024	14:35:46	558
9	Максим Федор...	Ольга Подуруе...	11.12.2024	20:31:18	1490
11	Андрей Федор...	Ольга Подуруе...	13.12.2024	12:23:51	654
13	Андрей Балун...	Владимир Зах...	13.12.2024	19:13:14	915
14	Александр Тел...	Александр По...	13.12.2024	19:38:15	1863
15	Надежда Прох...	Александр По...	13.12.2024	19:38:32	1058
16	Клавдия Тельн...	Александр По...	13.12.2024	19:38:48	725