

Санкт-Петербургский государственный политехнический университет  
Институт компьютерных наук и кибербезопасности  
**«Высшая школа программной инженерии»**

**КУРСОВОЙ ПРОЕКТ**

**Автоматизация работы магазина**

по дисциплине «Базы данных»

Выполнил

студент  
гр.5130904/10102

*<подпись>*

Приймак М. С.

Руководитель

*<подпись>*

Губенко Н. О.

«\_\_\_» \_\_\_\_\_ 202\_\_ г.

Санкт-Петербург  
2023

## Оглавление

Введение .....	3
Общая задача .....	3
Требования к базе данных .....	4
Требования к клиентскому приложению .....	5
Дополнительные требования .....	6
База данных .....	7
Взаимодействие с базой данных .....	9
Клиентское приложение .....	11
Используемые модули .....	11
Структура программы .....	13
Разделение возможностей пользователей .....	15
Многопоточная модель .....	16
Демонстрация работы .....	17
Заключение .....	22
Список использованных источников .....	23

## **Введение**

### **Общая задача**

В рамках базы данных PostgreSQL необходимо создать объекты в схеме вашего пользователя и написать клиентское приложение на базе компонентов ADO.NET, язык программирования C#.

Магазин занимается продажей со склада товаров оптом и в розницу. При этом менеджеры следят за тем, чтобы на складе постоянно были товары в наличии. Товары закупаются у посредника по установленной цене. В дальнейшем магазин продает их по другой цене. Продажи записываются в журнал продаж, где отмечается дата продажи, количество проданного товара и цена за единицу. Полученные от разницы в покупке и продаже товара деньги магазин тратит на различные статьи расхода. Данные о расходах отмечаются в журнале расходов и характеризуются датой, суммой и самой статьей расхода. В частности, каждый месяц магазин выплачивает работникам зарплату.

## Требования к базе данных

- 1) Контроль целостности данных, используя механизм связей
- 2) Операции модификации групп данных и данных в связанных таблицах должны быть выполнены в рамках транзакций.
- 3) Логика работы приложения должна контролироваться триггерами. В частности, должны быть реализованы следующие ограничения:
  - Не позволяет добавить в таблицу товаров товар, у которой не задана сумма.
  - Не позволяет добавлять расход, с суммой большей заданной.
  - Не позволяет изменять данные в таблице продаж задним числом от сегодняшней даты.
- 4) Все операции вычисления различных показателей (из требований к клиентскому приложению) должны реализовываться хранимыми процедурами.

## Требования к клиентскому приложению

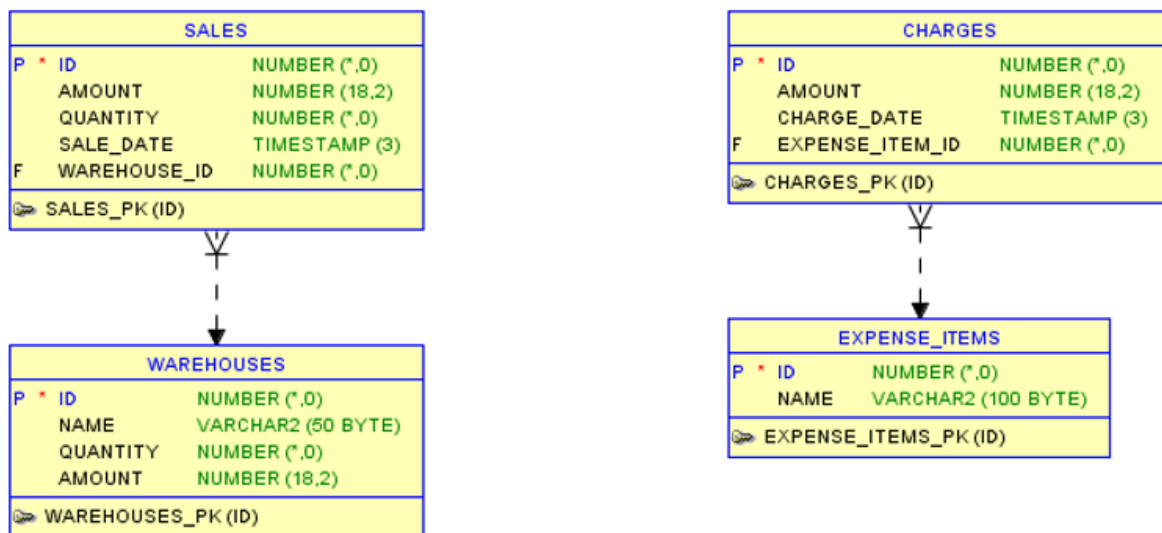
- 1) Необходимо реализовать интерфейсы для ввода, модификации и удаления справочников:
  - Товаров;
  - Статей расходов.
- 2) В главном окне приложения должны быть реализованы интерфейсы:
  - Журнала продаж с возможностью задавать товар, дату продажи, количество и цену за единицу.
  - Журнал расходов с возможностью задавать дату, сумму и статью расхода.
- 3) Необходимо реализовать возможность просмотра оператором следующих показателей:
  - Прибыль магазина за заданный месяц.
  - Пять самых доходных товаров за заданный интервал дат.

### **Дополнительные требования**

- а)** отдельный файл параметров подключения к БД, который задаются файлом (формат plain text file или ini config file или xml file);
- б)** авторизацию с хешированием паролей для пользователей (2 пользователя);
- в)** меню приложения содержит группы элементов: СПРАВОЧНИКИ, ЖУРНАЛЫ, ОТЧЕТЫ;
- г)** приложение должно обеспечивать ввод, редактирование, просмотр информации из БД;
- д)** используется пользовательский элемент вида GridView для обзора таблиц (представлений) БД;
- е)** GridView, который содержит внешние ключи, должен содержать нужные/необходимые атрибуты внешней таблицы;
- ж)** есть отдельная форма для работы с записью таблицы, содержащая эл-ты интерфейса ДОБАВИТЬ, ИЗМЕНИТЬ, УДАЛИТЬ, ЗАКРЫТЬ; открывается из GridView;
- з)** меню пункта ОТЧЕТЫ вызывает формы вывода отчетов (минимум 2-х) (выходной формат txt или xls или pdf);

## База данных

В течение курса была разработана соответствующая база данных для этой курсовой работы.



Обозначение всех столбцов:

Имя таблицы	Имя колонки	Расшифровка
warehouses		Таблица склада товаров
	id	Идентификатор записи
	name	Наименование
	quantity	Количество товара
	amount	Стоимость одной единицы товара
sales		Таблица продаж
	id	Идентификатор записи
	amount	Стоимость одной единицы товара
	quantity	Количество товара
	sale_date	Дата продажи
	warehouse_id	Товар
charges		Таблица расходов
	Id	Идентификатор записи
	amount	Сумма
	charge_date	Дата
	expense_item_id	Статья расхода
expense_items		Таблица статей расхода
	id	Идентификатор записи
	name	Наименование статьи

Помимо указанных в задании таблиц мною была создана еще одна: хранящая данные о пользователях приложения (в данном случае админ и «продавец»).

Имя таблицы	Имя колонки	Расшифровка
users	Id (int, primary key)	Идентификатор записи
	Name (varchar 255)	Имя (логин) пользователя
	Password (varchar 255)	Пароль (хеширован)



## Взаимодействие с базой данных

Для работы с базой данных через код мною были написаны хранимые процедуры, предназначенные для различных команд, а также созданы триггеры, регулирующие некоторые основополагающие свойства базы данных.

Вот некоторые из них:

1)

```
BEGIN
    UPDATE warehouses SET quantity=quantity_up WHERE id=id_up;
END;
```

Данная функция служит для обновления данных, а именно столбца quantity (количество), в таблице warehouses (склад товаров) по указанному пользователем программы id элемента.

2)

```
BEGIN
    RETURN QUERY
    SELECT * FROM sales ORDER BY warehouse_id;
END;
```

Служит для вывода полной таблицы sales (продажи) и сортирующей по внешнему ключу.

3)

```
BEGIN
    INSERT INTO warehouses (name, quantity, amount) VALUES (name_in, quantity_in, amount_in);
END;
```

Позволяет вставить новые данные в таблицу warehouses (склад товаров).

4)

```
BEGIN
    DELETE FROM warehouses WHERE id=id_de;
END;
```

Удаляет данные из таблицы warehouses (склад товаров) по введенному id элемента. Помимо удаления из указанной таблицы также удаляются все записи, связанные с этим элементом, из связанной с ней таблицы (то есть из таблицы sales удалятся все записи продаж этого товара).

5)

```
BEGIN
    RETURN QUERY
    SELECT
    (SELECT SUM(amount * quantity) FROM sales WHERE sale_date<month_to::date AND sale_date>month_from::date)
    -
    (SELECT SUM(amount) FROM charges WHERE charge_date<month_to::date AND charge_date>month_from::date)
    AS profit;
END;
```

Эта функция позволяет пользователю просмотреть общую прибыль магазина по всем товарам за указанный месяц.

## Клиентское приложение

### Используемые модули

Для реализации всех поставленных задач мной были выбраны следующие библиотеки (фреймворки):

- **WPF (Windows Presentation Foundation)** - это платформа пользовательского интерфейса, которая создает клиентские приложения для настольных компьютеров. Платформа разработки WPF поддерживает широкий набор компонентов для разработки приложений, включая модель приложения, ресурсы, элементы управления, графику, макет, привязки данных, документы и безопасность.

WPF является частью платформы .NET. WPF использует XAML расширяемого языка разметки приложений для предоставления декларативной модели программирования приложений.

Данный фреймворк, как уже стало ясно, используется для создания графического пользовательского интерфейса. В качестве языка разметки в нем используется XAML, а управление элементами разметки (назначение событий, управление состоянием) происходит в соответствующих классах контроллеров.

- **Npgsql** — это поставщик данных ADO.NET с открытым исходным кодом для PostgreSQL. Он позволяет программам, написанным на C#, Visual Basic, F#, получать доступ к серверу базы данных PostgreSQL. Он реализован на 100% коде C#, бесплатен и имеет открытый исходный код.

- **Aspose.Cells** — это надежная и многофункциональная библиотека, позволяющая разработчикам .NET создавать, загружать, отображать и экспортировать файлы электронных таблиц Excel в приложениях C#, ASP.NET и VB.NET. Используя эту библиотеку, вы можете читать, просматривать,

редактировать и писать таблицы Excel, а также выполнять сложные задачи, такие как расчет формул, управление диаграммами и сводными таблицами, а также форматирование текста в электронных таблицах.

Эта библиотека, а точнее ее подраздел Cells, используется для взаимодействия с Excel файлами через код языка C# **без необходимости** установленного ПО MS Office, в отличие от подобного фреймворка от Microsoft.

- **System** – стандартная библиотека языка C#, создающая фундаментальные и базовые классы, определяющие часто используемые типы значений и ссылочных данных, события и обработчики событий, интерфейсы, атрибуты и исключения обработки.

Конкретно мною задействованы модули, отвечающие за асинхронность, работу с текстом, таблицами, коллекциями и, конечно, модуль непосредственно с самим фреймворком WPF.

## Структура программы

Для реализации клиентской части данной курсовой работы был использован язык C#, подход ООП.

### Пояснение всех классов и устройства программы:

#### 1) Класс **App.xaml.cs**

Ключевой класс всей программы, являющийся «точкой входа» в программу, и который запускает главное окно приложения (экран авторизации пользователя).

#### 2) Класс **BaseInteraction.cs**

Предназначен для создания соединения с базой данных. В нем происходит лишь соединение, а дальше этот объект соединения используется по всей программе. Важно уточнить, что возможен лишь один такой объект соединения с конкретной базой данных.

#### 3) Класс **MainWindow.xaml.cs**

Этот файл наряду с `MainWindow.xaml` является частью одного класса, отвечающего за главное окно приложения. В файле с расширением `.xaml` находится разметка на языке расширений XAML, а в файле с расширением `.xaml.cs` уже находится контроллер данного окна, обработчик всех поступающих событий.

#### 4) Класс **SecondWindow.xaml.cs**

Данный класс устроен точно по такому же принципу, как и предыдущий – совместно с файлом `SecondWindow.xaml` реализуют класс основного окна взаимодействия пользователя и базы данных. Именно отсюда и происходит

дальнейшая работа с базой данных, посредством вызова соответствующих классов команд.

#### **5) Класс User.cs**

Отвечает за реализацию пользователей приложения.

#### **6) Перечисления (enum) ActiveCommand.cs, ActiveUser.cs и Tables.cs**

Данные перечисления созданы для удобства отслеживания различных состояний программы. Их назначения ясны из названий.

#### **7) Вспомогательные классы CommandData.cs и CommandResult.cs и Month.cs**

CommandData служит для удобства передачи значений, полученных с различных полей ввода информации от пользователя приложения, и дальнейшей их передачи на управляющие команды. CommandResult есть класс обертка для всех возвращаемых значений классов команд, взаимодействующих с базой данных. Month.cs нужен ли определения границ месяца, задаваемого пользователем.

#### **8) Классы команды из папки Commands**

Все классы из этой папки есть классы команд, расширяющие абстрактный класс Command, который объявляет общую для всех команд структуру (это есть применение паттерна «Шаблонный метод»). Каждый из классов реализует свой запрос к базе данных, например, классы, начинающиеся с ShowAll\*.cs, определяют запросы выборки данных (SELECT) для возможности просмотра пользователем соответствующей таблицы. А классы Insert.cs, Update.cs и Delete.cs делают возможным производить манипуляции с данными из таблиц.

## Разделение возможностей пользователей

По заданию мною были реализованы следующие роли пользователей: админ и «продавец».

**Админ** имеет доступ ко всему функционалу приложения: просмотр всех возможных таблиц, просмотр специально заготовленных функций таких как «прибыль за указанный месяц» и «5 самых доходных товара за указанный период», а также редактирование таблиц (вставка новых записей, их обновление и удаление).

**«Продавец»** имеет возможность лишь просматривать информацию из таблиц, а именно просмотр таблиц и заготовленных функций. Но не может их как-либо изменять.

## **Многопоточная модель**

Учитывая тот факт, что данная курсовая работа представляет из себя прототип реальных существующих систем, мною также был учтен момент многопользовательской работы с программой, то есть была использована многопоточная модель построения архитектуры.

Для реализации данной модели был выбран фреймворк WPF, который помимо реализации графического интерфейса сам внутри себя представляет многопоточную модель.

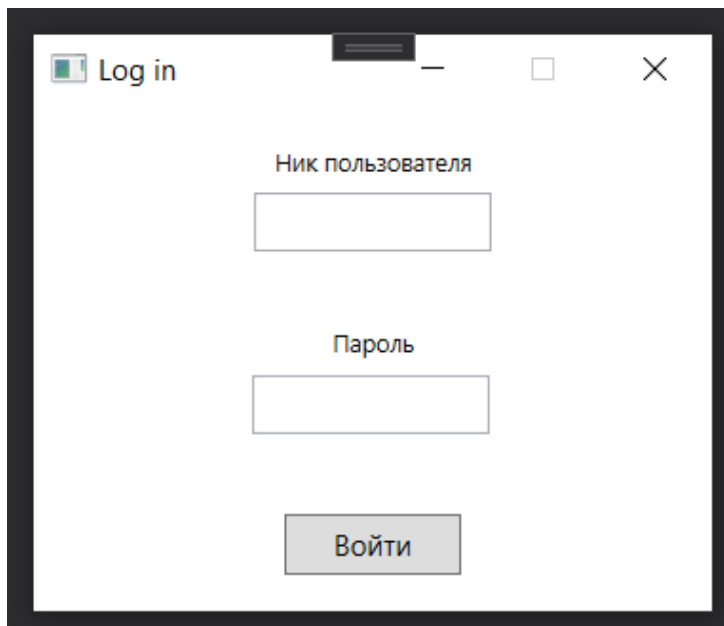
Как правило, приложения WPF начинаются с двух потоков: один для обработки отрисовки и другого для управления пользовательским интерфейсом. Поток визуализации эффективно выполняется незаметно для пользователя в фоновом режиме, тогда как поток пользовательского интерфейса получает входные данные, обрабатывает события, выводит изображение на экран и выполняет код приложения.

Поток пользовательского интерфейса создает очередь рабочих элементов внутри объекта с именем Dispatcher. Объект Dispatcher выбирает рабочие элементы на основе приоритетов и выполняет каждый из них до завершения. Каждый поток пользовательского интерфейса должен иметь хотя бы один объект Dispatcher, и каждый из объектов Dispatcher может выполнять рабочие элементы только в одном потоке.

Таким образом если в одну и ту же долю секунды к коду обратятся несколько пользователей, код не сломается, а поставит их запросы в очередь.



## Демонстрация работы



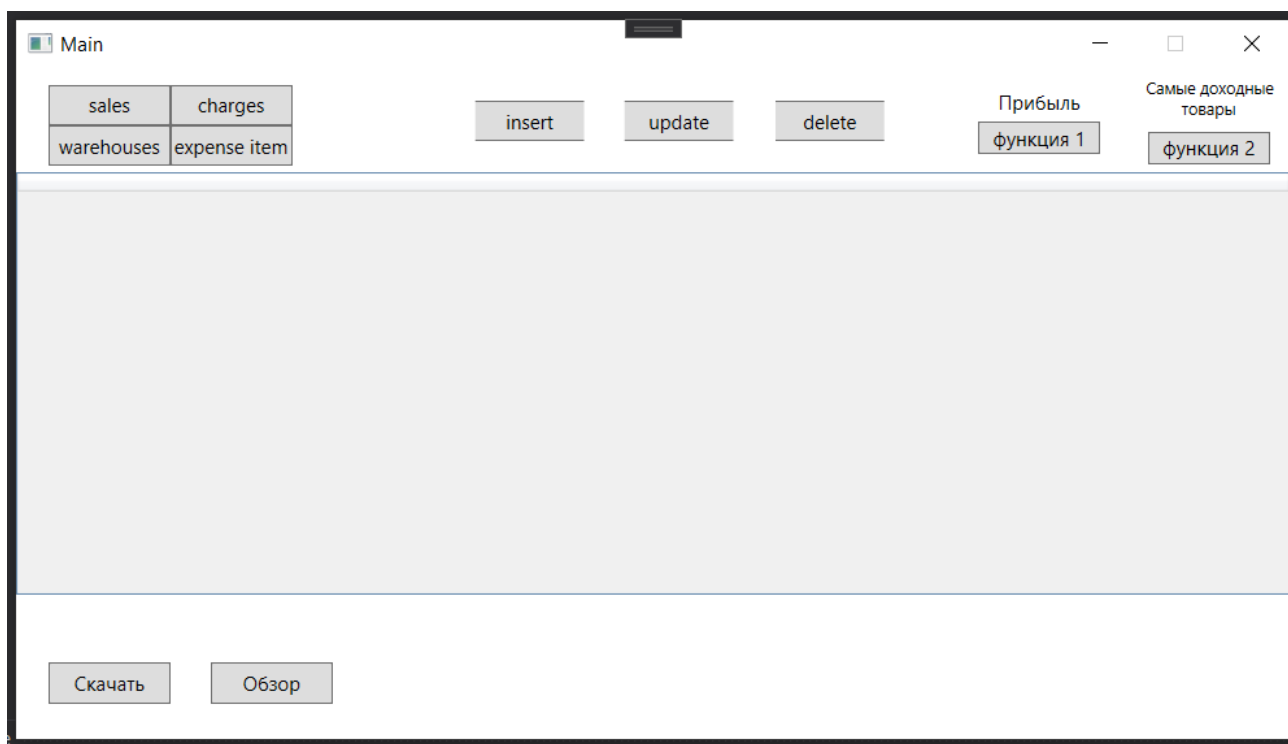
Log in

Ник пользователя

Пароль

Войти

Так выглядит окно входа. 2 роли: admin и seller, пароли такие же.



Main

sales	charges
warehouses	expense item

insert update delete

Прибыль функция 1

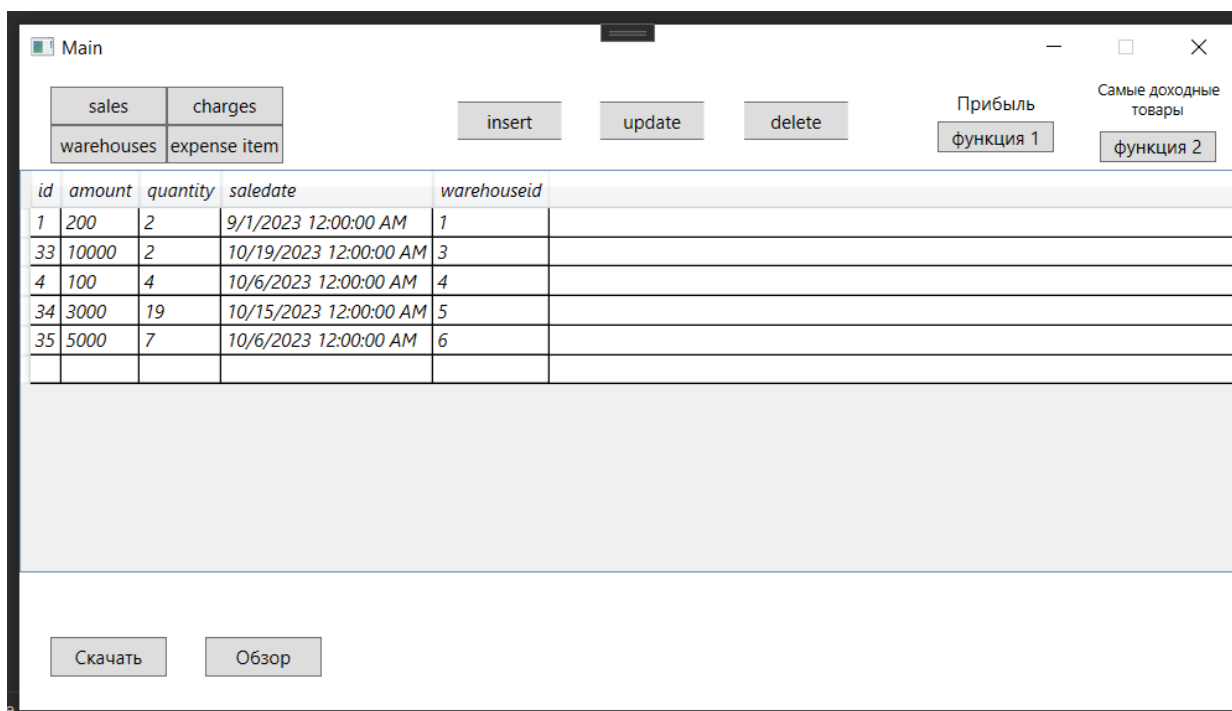
Самые доходные товары функция 2

Скачать Обзор

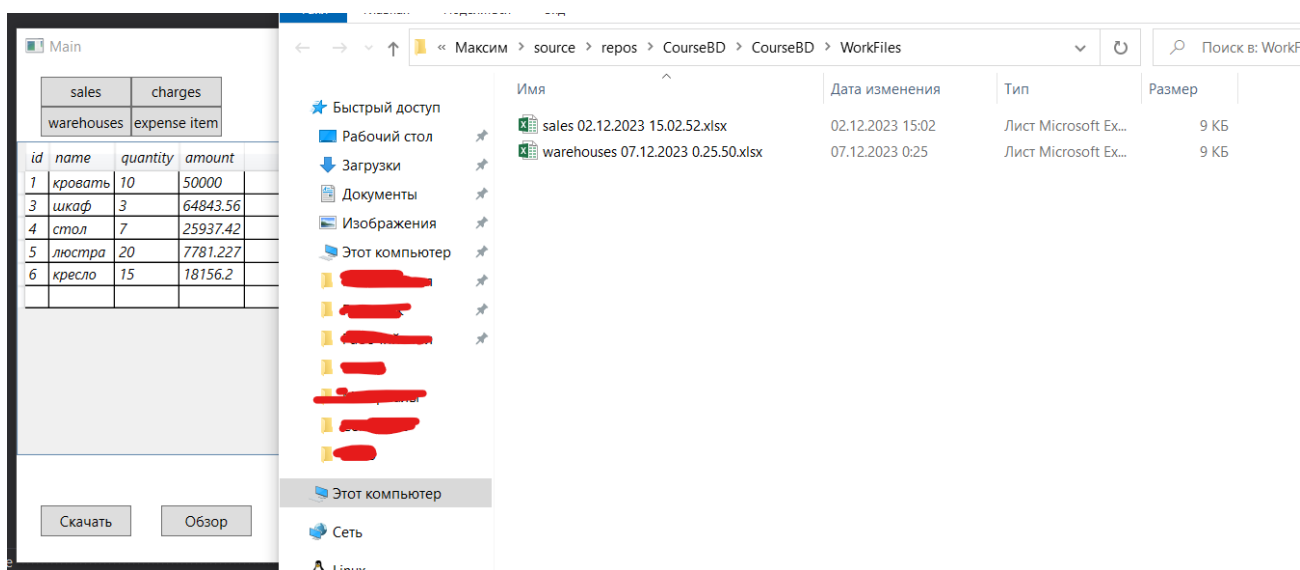
Это главное окно взаимодействия пользователя с базой данных (показан доступ admin, возможен весь функционал).



А так выглядит главное окно с доступом seller (возможен только просмотр данных).



Выше показана возможность просмотра таблиц и скачивания их (excel).



Сама процедура скачивания.

The screenshot shows a web application window titled 'Main' with a form for inserting, updating, or deleting records. The form includes buttons for 'insert', 'update', and 'delete'. There are also buttons for 'Прибыль' (Profit) and 'Самые доходные товары' (Most profitable goods). The form has input fields for 'id', 'quantity', 'name', and 'amount'. A dropdown menu labeled 'Выберите таблицу' (Select table) shows options for 'Товары' (Goods) and 'Статьи расходов' (Expense items). There are also buttons for 'функция 1' (Function 1), 'функция 2' (Function 2), and 'Очистить' (Clear). A 'Выполнить' (Execute) button is at the bottom.

Вот здесь показана возможность вставить новые записи в таблицы товаров (warehouses) и статей расходов (expense\_items).

Main

sales	charges
warehouses	expense item

insert    update    delete

Прибыль  
функция 1

Самые доходные товары  
функция 2

Выберите таблицу

Товары

Статьи расходов

id

name

Очистить

Выполнить

Возможность обновления данных.

Main

sales	charges
warehouses	expense item

insert    update    delete

Прибыль  
функция 1

Самые доходные товары  
функция 2

Выберите таблицу

Товары

Статьи расходов

id

Очистить

Выполнить

Удаление записей с последующим удалением из связанных таблиц.

The 'Main' window displays a profit view for the month of October. The interface includes a top menu bar with 'Main', a search bar, and window controls. Below the menu bar, there are two tabs: 'sales' and 'charges'. The 'sales' tab is active, showing a table with columns 'warehouses' and 'expense item'. To the right of the tabs are three buttons: 'insert', 'update', and 'delete'. Further right are two buttons: 'Прибыль' (Profit) and 'Самые доходные товары' (Most profitable goods). Below these buttons are two buttons: 'функция 1' (Function 1) and 'функция 2' (Function 2). A dropdown menu shows 'Октябрь' (October). The main area displays a table with columns 'profit' and an empty cell. The table contains one row with the value '111400'.

profit
111400

Здесь представлен просмотр прибыли за выбранный из списка месяц.

The 'Main' window displays a view of the 5 most profitable goods for a specific period. The interface includes a top menu bar with 'Main', a search bar, and window controls. Below the menu bar, there are two tabs: 'sales' and 'charges'. The 'sales' tab is active, showing a table with columns 'warehouses' and 'expense item'. To the right of the tabs are three buttons: 'insert', 'update', and 'delete'. Further right are two buttons: 'Прибыль' (Profit) and 'Самые доходные товары' (Most profitable goods). Below these buttons are two buttons: 'функция 1' (Function 1) and 'функция 2' (Function 2). Two date input fields are shown: '15.10.2023' and '19.10.2023', each with a calendar icon. A 'Выполнить' (Execute) button is located to the right of the date fields. The main area displays a table with columns 'id' and 'sum'. The table contains two rows: one with '34' and '57000', and another with '33' and '20000'.

id	sum
34	57000
33	20000

Просмотр 5 самых доходных товаров за указанный период (указываются конкретные дни).

## Заключение

В ходе данной курсовой работы мною были разработаны база данных и клиентское приложение. Для постройки базы данных была использована PostgreSQL, а для клиентского приложения язык C# совместно с фреймворком WPF.

База данных была создана используя механизм связи, все запросы к базе организованы в виде транзакций. Сами запросы хранятся в хранимых процедурах (функциях). Для регулировки данных таблиц были написаны триггеры.

Клиентское приложение спроектировано с возможной нагрузкой использования одновременно несколькими пользователями благодаря многопоточной структуре WPF. Команды (запросы) к базе данных реализованы каждая в виде класса (реализован паттерн «Шаблонный метод»). Также происходит разделение отображения окон и их логики. Помимо WPF также были использованы такие библиотеки как Aspose.Cells для работы с Excel файлами, библиотека Npgsql для взаимодействия с PostgreSQL и, конечно, стандартная библиотека System.

## Список использованных источников

- 1) <https://learn.microsoft.com/ru-ru/dotnet/csharp/>
- 2) <https://metanit.com/sharp/tutorial/?ysclid=lpu9as3h15362392776>
- 3) <https://learn.microsoft.com/ru-ru/dotnet/desktop/wpf/introduction-to-wpf?view=netframeworkdesktop-4.8>
- 4) <https://metanit.com/sharp/wpf/?ysclid=lpu9fiscdf104235168>
- 5) Язык программирования C# 7 и платформы .NET и .NET Core. Автор: Троелсен Э., Джепикс Ф. Дата выхода: 2018. Издательство: Компьютерное издательство "Диалектика". Количество страниц: 1330  
[https://coderbooks.ru/books/c\\_sharp/yazyk\\_programmirovaniya\\_c\\_7\\_i\\_platfor my\\_net\\_i\\_net\\_core\\_troelsen\\_2018/](https://coderbooks.ru/books/c_sharp/yazyk_programmirovaniya_c_7_i_platfor_my_net_i_net_core_troelsen_2018/)
- 6) <https://products.aspose.com/cells/net/>