











## ЗМІСТ

ВСТУП.....	7
1 ОПИС ПРЕДМЕТНОЇ ОБЛАСТІ.....	9
1.1 Аналіз предметної області.....	9
1.2 Мета й завдання проекту.....	9
1.3 Проектування бази даних за допомогою ER-діаграми.....	9
2 РОЗРОБКА БАЗИ ДАНИХ В MS ACCESS.....	13
2.1 Створення таблиць, заповнення даними та створення зв'язків між таблицями.....	13
2.2 Вивід даних в DataGridView.....	17
2.3 Створення обчислювального поля.....	18
2.4 Фільтрація даних.....	20
2.5 Створення запитів.....	21
2.6 Створення звітів.....	22
2.7 Розробка інтерфейсу.....	23
3 ПОСІБНИК КОРИСТУВАЧА.....	27
ВИСНОВКИ.....	35
ПЕРЕЛІК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ.....	36
ДОДАТОК А.....	37
ДОДАТОК Б.....	40
ДОДАТОК С.....	42

## ВСТУП

Будь-яка організація потребує своєчасного доступу до інформації. Цінність інформації в сучасному світі дуже висока. Роль розпорядників інформації в сучасному світі найчастіше виконують бази даних. Бази даних надають можливість безпечно зберігати інформацію в структурованому вигляді та своєчасний доступ до неї. Практично будь-яка сучасна організація має або потребує базу даних, що задовольняє ті чи інші потреби по зберіганню, управлінню та адмініструванню даних.

База даних (англ. database) – сукупність даних, організованих відповідно до концепції, яка описує характеристику цих даних і взаємозв'язки між їх елементами; ця сукупність підтримує щонайменше одну з областей застосування (за стандартом ISO/IEC 2382:2015). В загальному випадку база даних містить схеми, таблиці, подання, збережені процедури та інші об'єкти. Дані у базі організовують відповідно до моделі організації даних. Таким чином, сучасна база даних, крім саме даних, містить їх опис та може містити засоби для їх обробки [2].

При автоматизації бізнес процесів дуже часто виникають завдання, які не вирішують вже готові програми та бази даних. При цьому аналітична інформація показує, що навіть якщо використовувати складні і дорогі CRM-системи управління підприємством, отримати рішення, яке задовольняє керівництво компанії, буває просто не можливо.

Бази даних створюються спеціально для зберігання, обробки, проведення розрахунків, сортування, вибірки та подання будь-яких масивів даних з будь-яких критеріям [2].

Бази даних позбавлені великої нестачі файлів: з ними немає проблем зі спільним доступом до даних. Сценарій, який змінює файл в процесі своєї роботи, можуть запустити одночасно два чоловіки, і, якщо не вжити заходів для блокування файлу, то можуть виникнути проблеми. З базами даних таких

проблем не існує, тому що передбачено вирішення проблем спільного доступу на низькому рівні з максимальною ефективністю.

Як середовище для розробки прикладної програми для роботи з базою даних було обрано середовище об'єктно-орієнтованого програмування Visual Studio 2019 Windows Forms C#. Це середовище розробки має в складі безліч різноманітних засобів, службовців зручною та ефективною основою для розробки додатків [1].

Visual Studio 2019 Windows Forms C# може бути використаний скрізь, де потрібно доповнити існуючі застосування розширеним стандартом мови C# , підвищити швидкодію і додати призначеному для користувача інтерфейсу якості професійного рівня.

Система швидкої розробки ґрунтуються на технології візуального проектування і подієвого програмування. Програмування в подібних системах зводиться в основному до наглядного будівництва програм з готових компонентів.

Метою курсового проекту є отримання навичок створення автоматизованої інформаційної системи, створення таблиць, запитів, звітів, фільтрації, отримання навичок використання навігаційних засобів програми Visual Studio 2019 Windows Forms C#.



# **1 ОПИС ПРЕДМЕТНОЇ ОБЛАСТІ**

## **1.1 Аналіз предметної області**

Завданням курсового проекту є розробка бази даних, яка охоплює предметну область “AIC для обліку роботи кабельного телебачення (надання послуг населенню)”.

При проектування бази даних необхідно зберігати інформацію про Клієнтів, працівників, заказів, та робіт за замовлення на локальному сервері.

Телебачення – виробництво аудіовізуальних програм та передач або комплектування (пакування) придбаних аудіовізуальних програм та передач і їхнього поширення незалежно від технічних засобів розповсюдження [2].

## **1.2 Мета й завдання проекту**

Метою курсового проекту є:

- Збирання даних або первинної інформації, обробка даних і отримання результатної інформації. Тому потрібно створити базу даних, за допомогою якої можна було б використовувати всю інформацію. Передача даних на сервер.
- Основне завдання при розробці бази даних складається в можливості експлуатації й коректної роботи з даними в ній.

## **1.3 Проектування бази даних за допомогою ER-діаграми**

Було побудовано ER-діаграму (див. Рисунок 1), в якій позначені: таблиці, поля, ключові поля та зв'язок між таблицями.

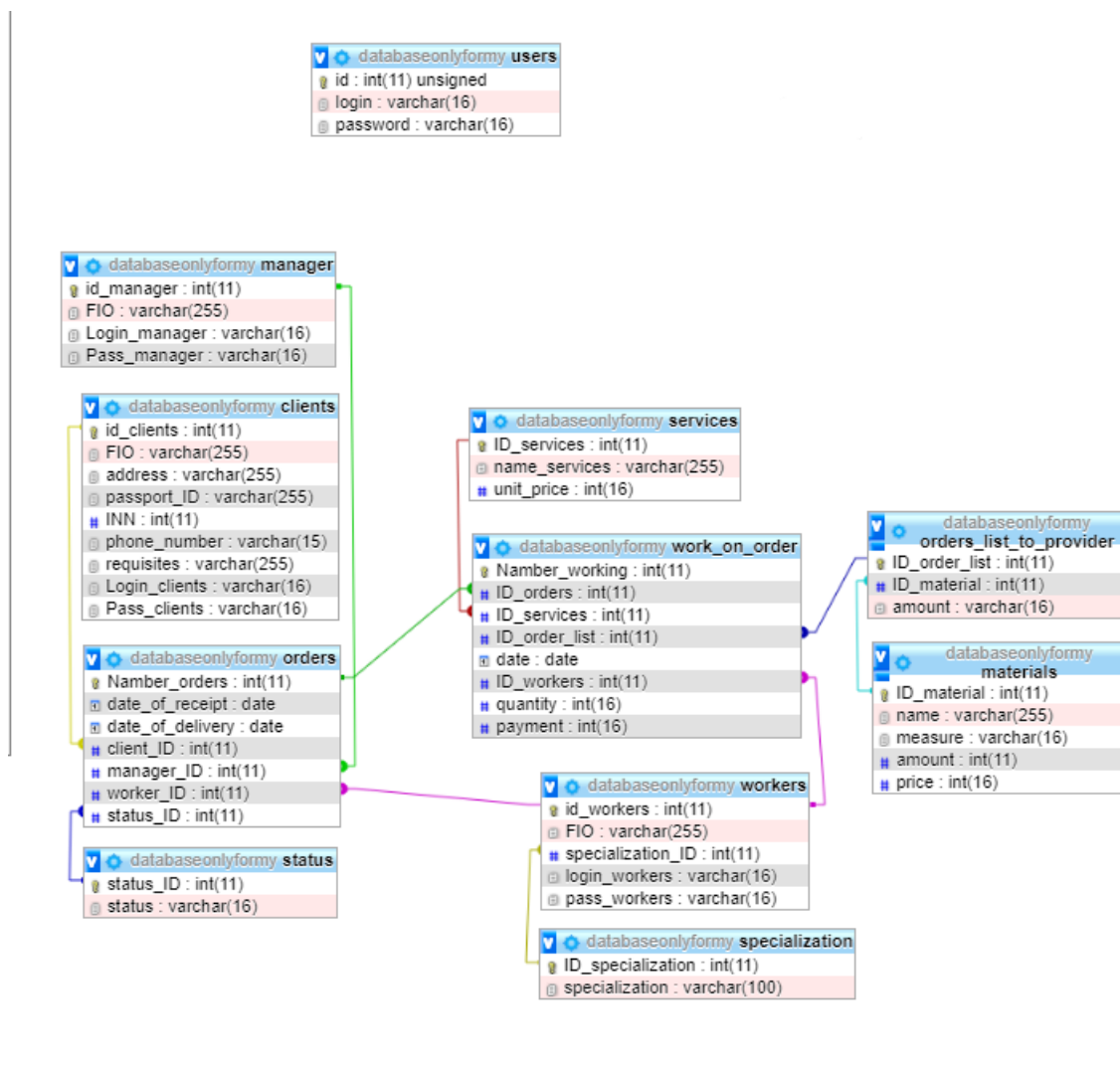


Рисунок 1 – ER-діаграма

Усі поля таблиць бази даних з їх описом (тип даних, маски, підстановки, тощо) можна переглянути на таблицях 1 – 11.

Таблиця 1 – поля таблиці «матеріали»/materials

Назва поля	Тип даних	Примітка
ID_материала ID materials	int(16)	Ключове поле
Наименование name	varchar(255)	-
Единица измерения measure	varchar(16)	-
Количество amount	int(16)	-

Цена prise	int(16)	-
Валюта	varchar(16)	За замовчуванням

Таблиця 2 – поля таблиці «менеджеры»/managers

Назва поля	Тип даних	Примітка
ID_менеджера ID_manager	int(16)	Ключове поле
ФИО FIO	varchar(255)	-
Логин login_manager	varchar(16)	-
Пароль pass_manager	varchar(16)	-

Таблиця 3 – поля таблиці «клиенты»/clients

Назва поля	Тип даних	Примітка
ID_клиента id_client	int(16)	Ключове поле
ФИО FIO	varchar(255)	-
Место_проживания adres	varchar(255)	-
Номер_паспорта passport id	varchar(255)Null	-
ИНН INN	varchar(255)Null	-
Номер_телефона phone nomber	varchar(255)	-
Реквизиты reqvesits	varchar(255)	-
Логин login_client	varchar(16)	-
Пароль pass_client	varchar(16)	-

Таблиця 4 – поля таблиці «пользователи»/users

Назва поля	Тип даних	Примітка
ID	int(16)	Ключове поле
Логин login	varchar(16)	-
Пароль password	varchar(16)	-

Таблиця 5 – поля таблиці «заказ»/order

Назва поля	Тип даних	Примітка
ID_заказа Number order	int(16)	Ключове поле
Дата_принятия_заказа data of receipt	Date	-
Дата_сдачи_заказа data of deliver	Date(Null)	-
id_клиента client ID	int(16)	Зв'язок з таблицею «клиенты»
id_менеджера manager ID	int(16)	Зв'язок з таблицею «менеджеры»
id_специалиста worked ID	int(16)	Зв'язок з таблицею «специалисты»
id_статуса status ID	int(16)	Зв'язок з таблицею «статус заказа»

Таблиця 6 – поля таблиці «работы по заказу»/work\_on\_order

Назва поля	Тип даних	Примітка
ID_работ NOMber_working	int(16)	Ключове поле
id_заказа ID_order	int(16)	Зв'язок з таблицею «заказ»
id_услуги ID servises	int(16)	Зв'язок з таблицею «услуги»
id_списка ID_order_list	int(16)	Зв'язок з таблицею «список_материалов_по_заказу»
id_специалиста ID worked	int(16)	Зв'язок з таблицею «специалисты»
Дата Date	varchar(10)Null	-
Количество quantity	int(16)	-
Цена payment	int(16)	Обчислювальне поле
Валюта	varchar(16)	За замовчуванням

Таблиця 7 – поля таблиці «специализация»/spechialization

Назва поля	Тип даних	Примітка
ID_специализации ID spechialization	int(16)	Ключове поле
Специализация spechialization	varchar(255)	-

Таблиця 8 – поля таблиці «спеціалісти»/worked

Назва поля	Тип даних	Примітка
ID_спеціаліста id_worked	int(16)	Ключове поле
ФІО FOI	varchar(255)	-
id_спеціалізації specialization ID	int(16)	Зв'язок з таблицею «спеціалізація»
Логін login_worked	varchar(16)	-
Пароль pass_worked	varchar(16)	-

Таблиця 9 – поля таблиці «услуги»/services

Назва поля	Тип даних	Примітка
ID_услуги ID_services	int(16)	Ключове поле
Наименование name	varchar(255)	-
Ценовая_политика unit_price	int(16)	-
Валюта	varchar(16)	За замовчуванням

Таблиця 10 – поля таблиці «статус заказа»/status

Назва поля	Тип даних	Примітка
ID_статуса status ID	int(16)	Ключове поле
Статус status	varchar(255)	-

## 2 РОЗРОБКА БАЗИ ДАНИХ В MS ACCESS

### 2.1 Створення таблиць, заповнення даними та створення зв'язків між таблицями

Перед початком роботи у Visual Studio 2019 Windows Forms C#, створюємо базу даних з таблицями на локальному сервері MAMP - призначена для створення, редагування та зберігання даних і структур таблиці БД на локальному сервері. Для початку треба завантажити та встановити програму MAMP. Після цього необхідно запустити сам сервер та натиснувши на “open WEBstart Page” перейти на сторінку браузера ( Рисунок 2 ) , де буде видана важлива інформація, а саме Host, Port, User, Password та посилання на сторінку phpMyAdmin, тобто на сторінку створення самої БД.

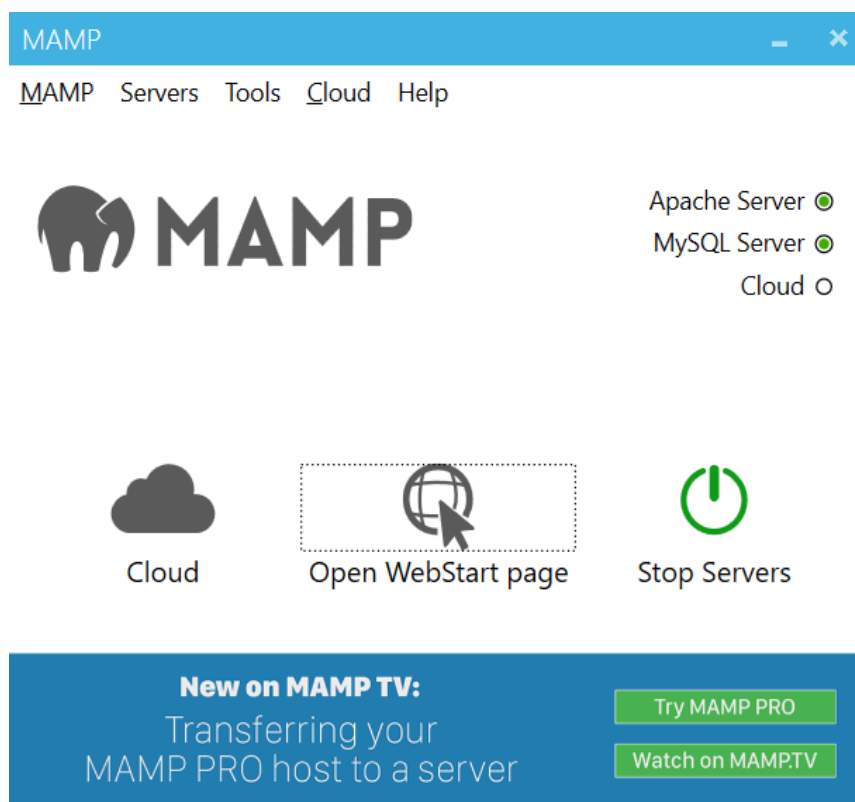


Рисунок 2 – Створення локального сервера

У вікні браузера, яке відкриється після відкриття сторінки phpMyAdmin, необхідно створити свою базу даних. Для цього натискаємо на кнопку “Создать

БД”, яка знаходиться в лівій частини екрана. В цієї ж частині екрану йде перелік всіх баз даних, а саме тих, що створились автоматично (“information\_schema”, “mysql”, “performance\_schema”, “sys”), та тих, що були створені користувачем (у нашому випадку це “bd\_tv\_services”) (Рисунок 3).

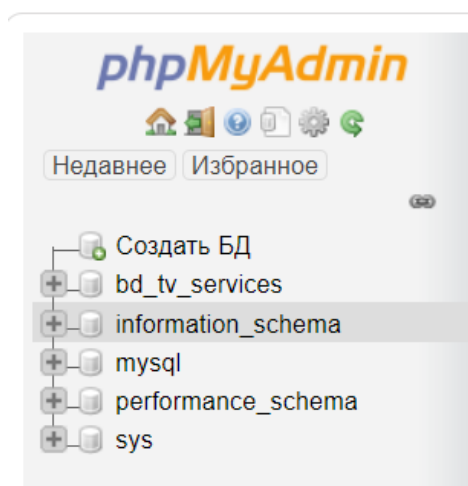


Рисунок 3 – Перелік БД на локальному сервері

Після створення БД необхідно її наповнити таблицями. Для створення таблиць необхідно спочатку натиснути на новостворену БД та в формі, яка виникає автоматично перед користувачем, ввести назву таблиці та змінити кількість стовбчиків (якщо необхідно), після цього натиснути на кнопку вперед. Відкриється конструктор таблиці, де необхідно налаштувати стовбчики таблиці, а саме: дати назву стовбчику, оголосити тип даних, обмежити кількість символів, що вводяться, та створити Ключове поле натиснув на пункт А\_І. Виконавши усі необхідні налаштування натискаємо на кнопку “Сохранить”.

Приклад конструктора таблиць зображено на Рисунку 4.

Сервер: localhost:3306 » База данных: vv » Таблица: vv

Имя таблицы: vv    Добавить 1 поле(я)    Вперёд

Имя	Тип	Длина/Значения	По умолчанию	Сравнение	Атрибуты
	INT		Нет		
	INT		Нет		
	INT		Нет		
	INT		Нет		

Комментарии к таблице:    Сравнение:    Тип таблиц: InnoDB

Определение разделов (PARTITION):

Критерий: ( Выражение или перечень )

Разделы:

Предпросмотр SQL    Сохранить

Консоль

Рисунок 4 – Конструктор таблиц

Створивши всі необхідні таблиці необхідно встановити зв'язок між ними. Для цього заходимо необхідну таблицю, натискаємо на пункт “Структура->Зв'язки” та налаштовуємо зв'язок між колонками таблиць. Приклад створення зв'язків між таблицями зображено на Рисунку 5.



Обзор Структура SQL Поиск Вставить Экспорт Импорт Привилегии Ещё

Структура таблицы Связи

### Ограничения внешнего ключа

Действия	Свойства ограничения	Столбец	Ограничение внешнего ключа (INNODB)		
			База данных	Таблица	Столбец
	Ограничения внешнего кл				
ON DELETE	RESTRICT	+ Добавить столбец			
ON UPDATE	RESTRICT				

+ Добавить ограничение

Предпросмотр SQL Сохранить

### Индексы

Действие	Имя индекса	Тип	Уникальный	Упакован	Столбец	Уникальных элементов	Сравнение	Null	Комментарий
Изменить Удалить	PRIMARY	BTREE	Да	Нет	ID_менеджера	2	A	Нет	

Создать индекс для 1 столбца/ов Вперёд

Рисунок 5 – Створення зв'язків між таблицями

Всі таблиці, їх конструктори та зв'зки представлені у ДОДАТКУ А.

## 2.2 Вивід даних в DataGridView

Для того щоб вивести дані в поле DataGridView необхідно створити клас, в якому буде знаходитись код для підключення MySQLServer, тобто нашої БД [1].

```
using System; using MySql.Data.MySqlClient;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace WindowsFormsApp1
{
    class DataBase
    {
        MySqlConnection connection = new MySqlConnection("server=localhost; port=3306;
User=root; Password=root; database=bd_tv_services");

        public void openConection()
        {
            if (connection.State == System.Data.ConnectionState.Closed)
                connection.Open();
        }

        public void closeConection()
        {
            if (connection.State == System.Data.ConnectionState.Open)
                connection.Close();
        }

        public MySqlConnection getConnection()
        {
            return connection;
        }
    }
}
```

Рисунок 6 — Підключення до локальної БД

Після цього на необхідній формі треба створити компонент DataGridView. Для цього відкриваємо необхідну форму та на панелі елементів знаходимо DataGridView. З самого початку цей компонент буде пустим. Для того, щоб дані передавалися необхідно створити подію, яка буде автоматично спрацьовувати під час завантаження самої форми. В панелі свойств форми переходимо на вкладку події та шукаємо подію “Load”, двічі натискаємо на неї та починаємо писати код.

Спочатку треба підключити бібліотеку `using MySql.Data.MySqlClient`, яка потрібна для роботи с деякими подіями, а саме виконання команд на

підключення БД та заповнення компонента DataGridView даними.

Після цього необхідно оголосити private класи:

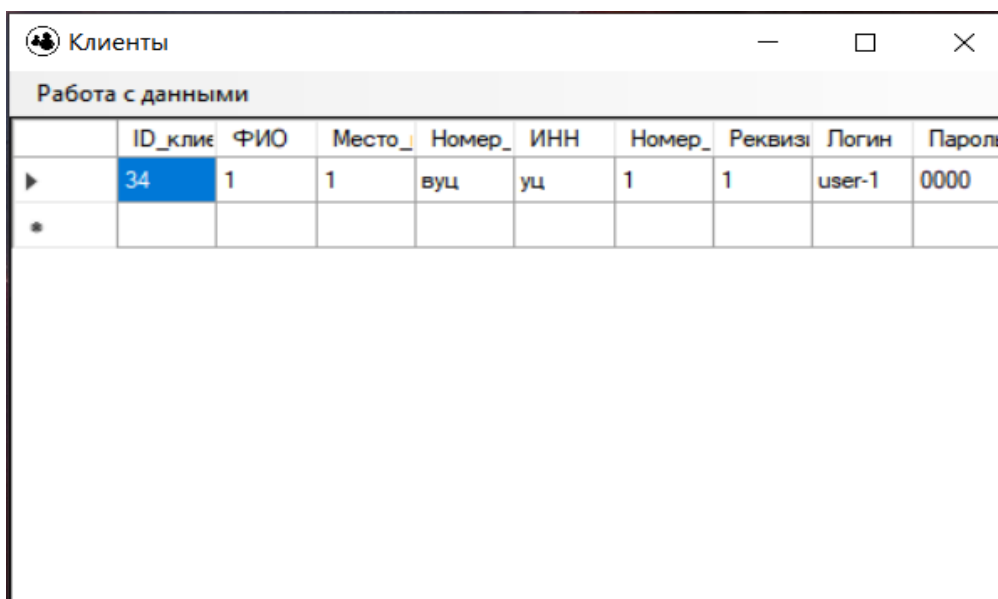
```
private DataBase db = null;
private DataTable table = null;
private MySqlDataAdapter adapted = null;
```

Після необхідно заповнювати DataGridView даними при завантаженні форми.

```
private void clientsForm_Load(object sender, EventArgs e)
{
    db = new DataBase();
    db.openConnection();
    adapted = new MySqlDataAdapter("SELECT * FROM `клиенты`",
    db.getConnection());
    table = new DataTable();

    adapted.Fill(table);
    dataGridView1.DataSource = table;
    db.closeConection();
}
```

Результат зображено на Рисунку — 8.



	ID_клие	ФИО	Место_	Номер_	ИНН	Номер_	Реквизи	Логин	Пароль
▶	34	1	1	вуц	уц	1	1	user-1	0000
*									

Рисунок 7 — Компіляція форми

## 2.3 Створення обчислювального поля

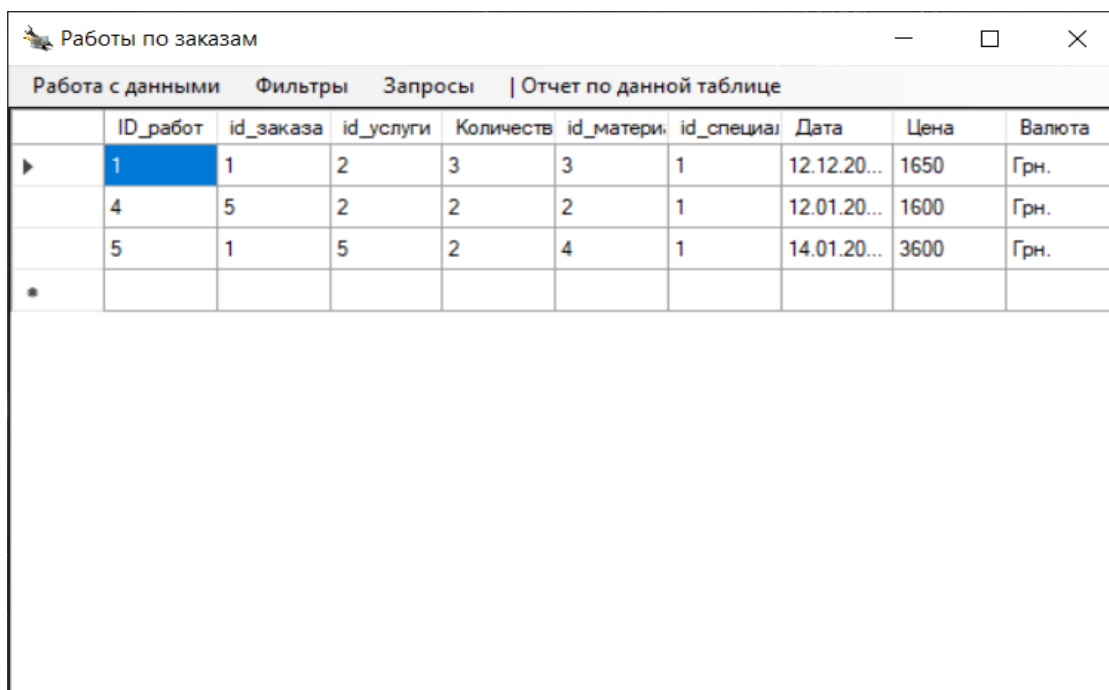
Обчислювальні поля призначені для показу даних, які автоматично обчислюються в процесі роботи програми, використовуючи одне або кілька полів набору даних. Весь процес автоматичного розрахунку представлений на Рисунку

-9, де необхідно при будь якій зміні поля автоматично заповняти певну колонку даних.

```

db = new DataBase();
if (dataGridView1.SelectedRows.Count != 1)
{
    MessageBox.Show("Выберите одну строку", "Внимание!");
    return;
}
int index = dataGridView1.SelectedRows[0].Index;
var price_services = dataGridView1.Rows[index].Cells[2];
var price_material = dataGridView1.Rows[index].Cells[4];
var price_koll = dataGridView1.Rows[index].Cells[3];
int material = 1;
int servis = 1;
if ((int)price_services.Value == 1)
    servis = 1200;
if ((int)price_services.Value == 2)
    servis = 4000;
if ((int)price_services.Value == 3)
    servis = 500;
if ((int)price_services.Value == 4)
    servis = 3000;
if ((int)price_services.Value == 5)
    servis = 4500;
if ((int)price_services.Value == 6)
    servis = 100;
if ((int)price_material.Value == 8)
    material = 50;
if ((int)price_material.Value == 9)
    material = 50;
if ((int)price_material.Value == 10)
    material = 10;
if ((int)price_material.Value == 11)
    material = 50;
if ((int)price_material.Value == 12)
    material = 50;
if ((int)price_material.Value == 13)
    material = 50;
if ((int)price_material.Value == 14)
    material = 200;
var price = material * (int)price_koll.Value + servis;
db.openConnection();
command_insert = new MySqlCommand("INSERT INTO `работы_по_заказу` (`id_заказа`,
`id_услуги`, `Количество_материалов`, `id_материала`, `id_специалиста`, `Дата`, `Цена`)
VALUES (@ff1, @ff2, @ff3, @ff4, @ff5, @ff6, @ff7)", db.getConnection());
command_insert.Parameters.Add("@ff1", MySqlDbType.Int16).Value =
dataGridView1.Rows[index].Cells[1].Value;
command_insert.Parameters.Add("@ff2", MySqlDbType.Int16).Value =
dataGridView1.Rows[index].Cells[2].Value;
command_insert.Parameters.Add("@ff3", MySqlDbType.Int16).Value =
dataGridView1.Rows[index].Cells[3].Value;
command_insert.Parameters.Add("@ff4", MySqlDbType.Int16).Value =
dataGridView1.Rows[index].Cells[4].Value;
command_insert.Parameters.Add("@ff5", MySqlDbType.Int16).Value =
dataGridView1.Rows[index].Cells[5].Value;
command_insert.Parameters.Add("@ff6", MySqlDbType.Date).Value =
dataGridView1.Rows[index].Cells[6].Value;
command_insert.Parameters.Add("@ff7", MySqlDbType.Int16).Value = price;

```



	ID_работ	id_заказа	id_услуги	Количеств	id_матери	id_специал	Дата	Цена	Валюта
▶	1	1	2	3	3	1	12.12.20...	1650	Грн.
	4	5	2	2	2	1	12.01.20...	1600	Грн.
	5	1	5	2	4	1	14.01.20...	3600	Грн.
•									

Рисунок 9 – Створення обчислювального поля Calculated

При будь якій зміні даних це поле буде автоматично змінено.

## 2.4 Фільтрація даних

Для створення фільтрації необхідно розмістити на формі компоненти menuStrip та декілька компонентів, що будуть служити фільтрами, при натисканні на які, буде відбуватися фільтрація. (Інтерфейс системи фільтрації див. рисунок 8, фільтрацію див рисунок 10).

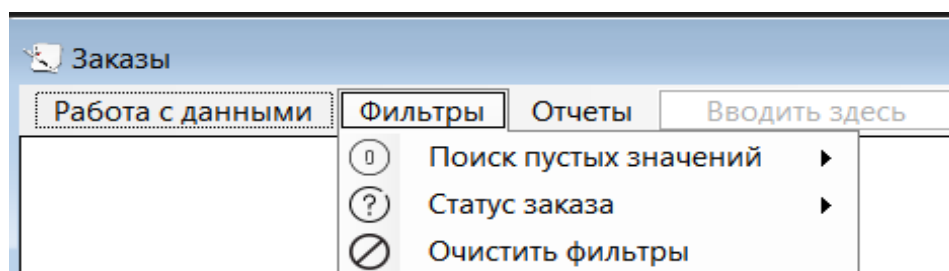


Рисунок 9 – Інтерфейс фільтрації даних

В Пункті фільтр вибрати один з представлених фільтрів, за якими буде відбуватися фільтрація. При натисканні на ці фільтри буде відбуватися подія, код якої представлений на Рисунку 10.

```
private void датасдачизаказаToolStripMenuItem_Click(object sender, EventArgs e)
{
    table = new DataTable();
    db = new DataBase();
    table.Clear();
    db.openConnection();
    adapted = new MySqlDataAdapter(
        "SELECT * FROM `заказ` WHERE `Дата_сдачи_заказа` IS NULL",
        db.getConnection());
    adapted.Fill(table);
    dataGridView1.DataSource = table;
    db.closeConnection();
}
```

Рисунок 10– Код фільтрації даних



Работа с данными    Фильтры    Отчеты							
	ID_заказа	Дата_принят	Дата_сдачи_	id_клиента	id_менеджера	id_специалиста	id_статуса
▶	14	14.12.2020		38	2	1	2
	15	01.12.2020		39	1	1	1

Рисунок 11 – Фільтрація даних

## 2.5 Створення запитів

Запити - найважливіший інструмент будь-якої системи управління базами даних (СУБД). Вони служать для вибірки певних записів з бази, оновлення таблиць і включення в них нових записів. Найчастіше запити використовують для фільтрації конкретних груп записів, що задовольняють певній умові. Крім того, вони ще і дозволяють комбінувати інформацію що зберігається в різних таблицях, забезпечуючи уніфікований вид, пов'язаним елементам даних [2].

Розробка запитів нічим не відрізняється від створення фільтрів бо ці два компонента роблять пошук даних на основі SQL запита.

Всі результати запитів та код, що їх виконує представлен у ДОДАТКУ Б.

## 2.6 Створення звітів

Звіт (report) - це об'єкт бази даних, який використовується для виведення на екран, до друку або у файл структурованої інформації. Звіти дозволяють отримати з таблиць або запитів бази даних необхідну інформацію і представити її у зручному для сприйняття вигляді [2].

При друку таблиць і запитів інформація видається практично в тому вигляді, в якому зберігається. Часто виникає необхідність представити дані у вигляді звітів, які мають традиційний вигляд і легко читаються. Докладний звіт включає всю інформацію з таблиці або запиту, але містить заголовки і розбитий на сторінки із зазначенням колонтитулів.

Для створення звітів в компоненті menuStrip розробляємо новий компонент, який називаємо “Отчеты” (Код події натискання представлений на Рисунку 12). Через те, що БД підключена примусова створити звіт завдяки програмному коду не вийде, тому необхідно користуватися вже готовим рішенням phpMyAdmin де і знаходиться наша БД.

```
private void полныйОтчетToolStripMenuItem_Click(object sender, EventArgs e)
{
    WebBrowser web = new WebBrowser();
    System.Diagnostics.Process.Start(@"chrome.exe", "http://localhost/MAMP/index.php?
page=phpmyadmin&language=English");
}
```

Рисунок 12 — Подія для відкриття браузеру та формування звіту

При натисканні на цей компонент перед користувачем відкривається вікно

браузера Google Chrome з новою сторінкою. На цій сторінці необхідно вибрати необхідний формат формування звіту, після цього натиснуту кнопку, що сформує необхідним нам звіт (Див. Рисунок 13)

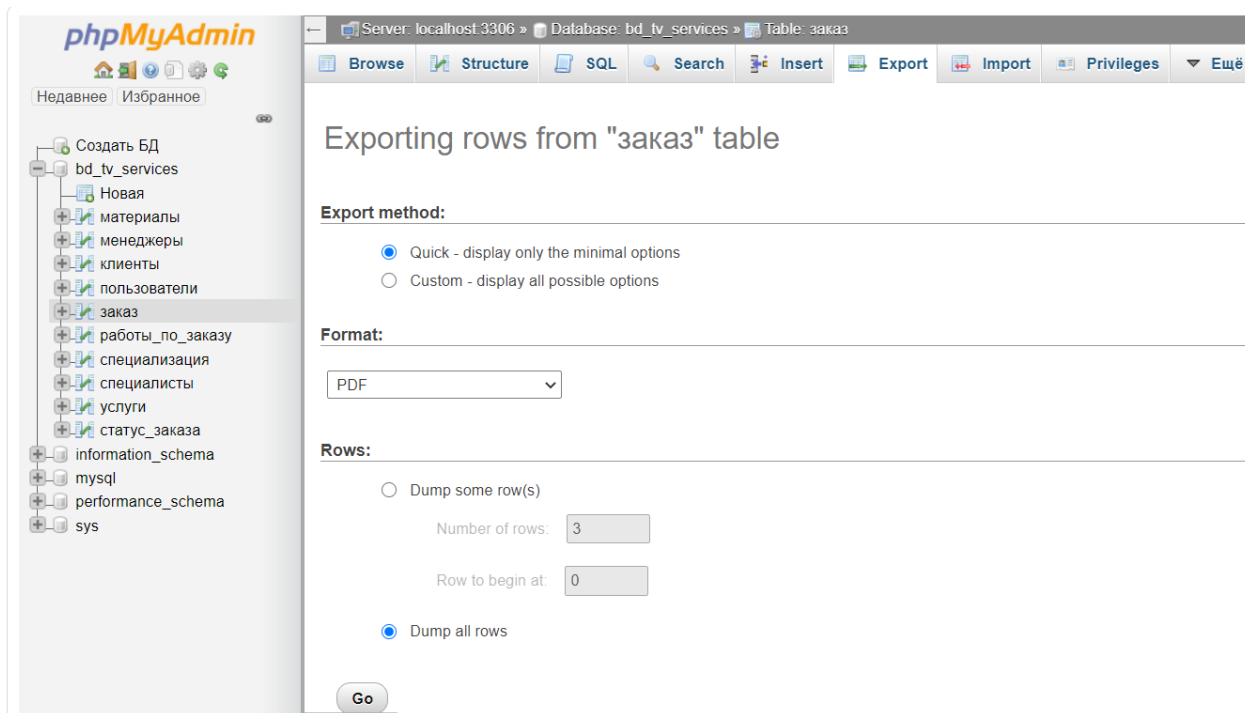


Рисунок 13 — формування звіту на сторінці phpMyAdmin

## 2.7 Розробка інтерфейсу

Форми є основою додатків Visual Studio 2019. Створення інтерфейсу в додатку полягає в додаванні у вікно форми елементів об'єктів Visual Studio 2019, що називаються компонентами. Компоненти Visual Studio 2019 розташовуються на палітрі компонентів, виконаної у вигляді багатосторінкового блокнота (PageControl - багатосторінкове вікно, дозволяє створювати сторінки в стилі Windows, управляти закладками чи іншими органами управління для економії місця на робочому столі). Важлива особливість Visual Studio 2019 полягає в тому, що він дозволяє створювати власні компоненти і налаштовувати палітру компонентів, їх подій, а також створювати різні версії палітри компонентів для



різних проектів.

На головній формі розташовані компонент MainMenu – основне меню, за допомогою якого можна переходити на вкладки: Файл, Таблицы для работы, Справочник, Справка; натискаючи на котрі можна відкривати інші форми: Заказы, работы по заказам, клиенты, менеджеры, специалисты, специализация, материалы, услуги и статусы заказов. Головна форма представлена на рисунку 14.

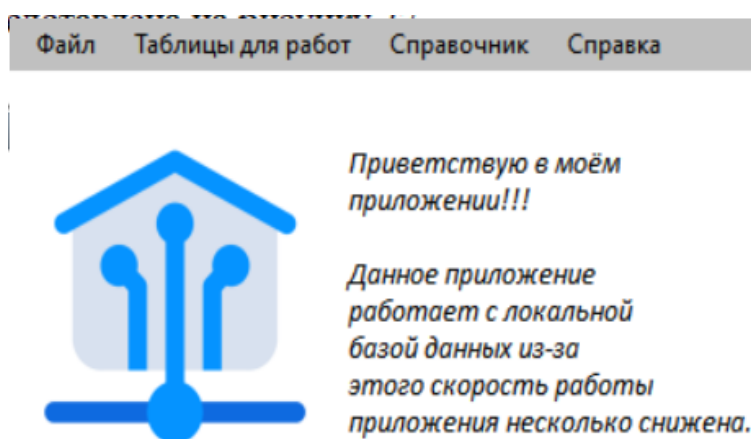


Рисунок 14 – Форма головного вікна програми

## 2.8 Розробка пунктів меню «Информация о компании», «Переход между формами», «Выход»

Для створення пункту "Про розробника" були використані компоненти Lebal , Panel (Див. Рисунку 15).

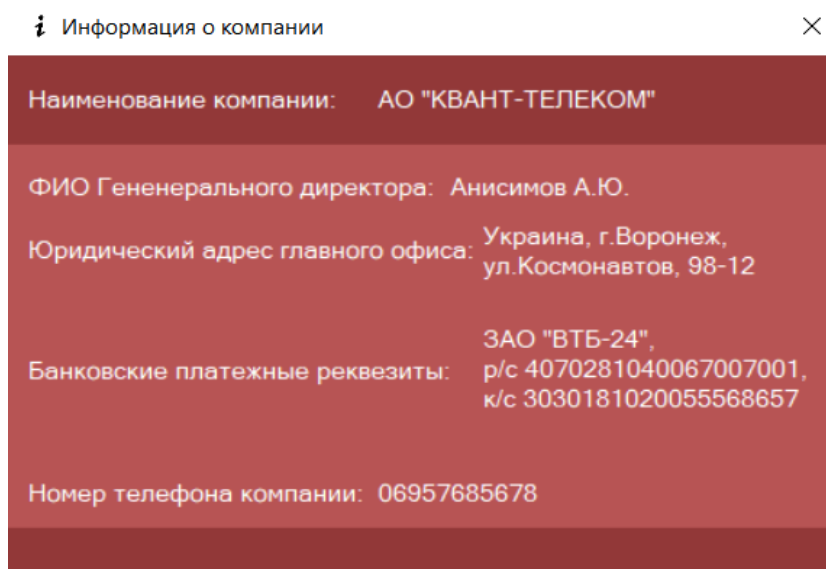


Рисунок 15 – Пункт меню «Информация о компании»

Для того, щоб створити пункт меню «Вихід», необхідно написати код. Це представлено на Рисунку 16. Інтерфейс вкладки представлений на рисунку 17.

```
private void выходToolStripMenuItem_Click(object sender, EventArgs e)
{
    if (MessageBox.Show("Подтвердить выход из приложения?", "Выход из приложения",
        MessageBoxButtons.YesNo) == System.Windows.Forms.DialogResult.Yes)
        Application.Exit();
}
```

Рисунок 16– Код пункту меню «Вихід»

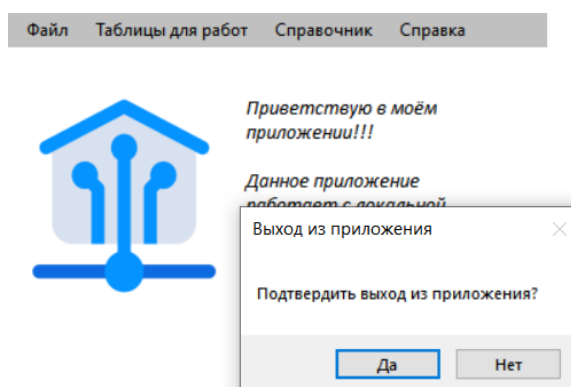


Рисунок 17 – Пункт меню «Вихід»

Перехід між формами здійснюється за допомогою фрагмента коду, що представлений на Рисунку — 18.

```
this.Hide();  
LoginForm loginForm = new LoginForm();  
loginForm.Show();
```

Рисунок 18 — Код переходу між формами

### 3 ПОСІБНИК КОРИСТУВАЧА

Після запуску виконавчого файлу, з'явиться форма загрузки програми, яка виконує загрузку самої програми (Див. Рисунок 19).

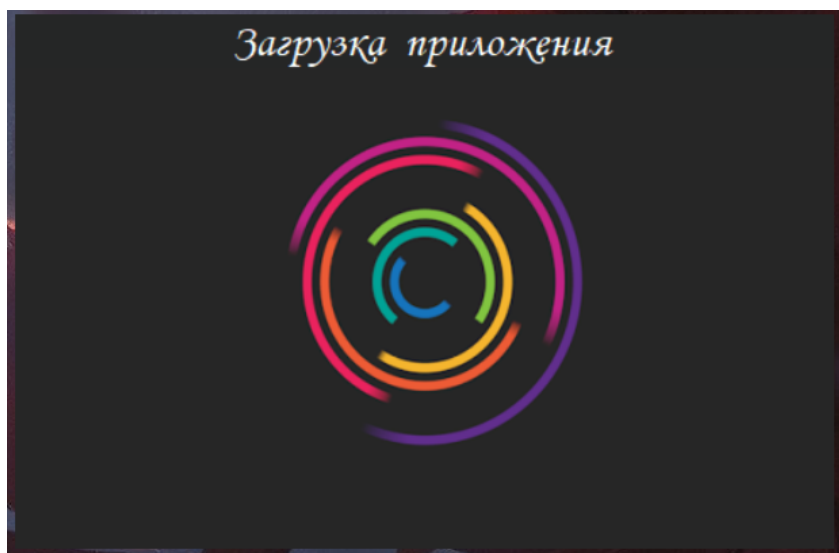
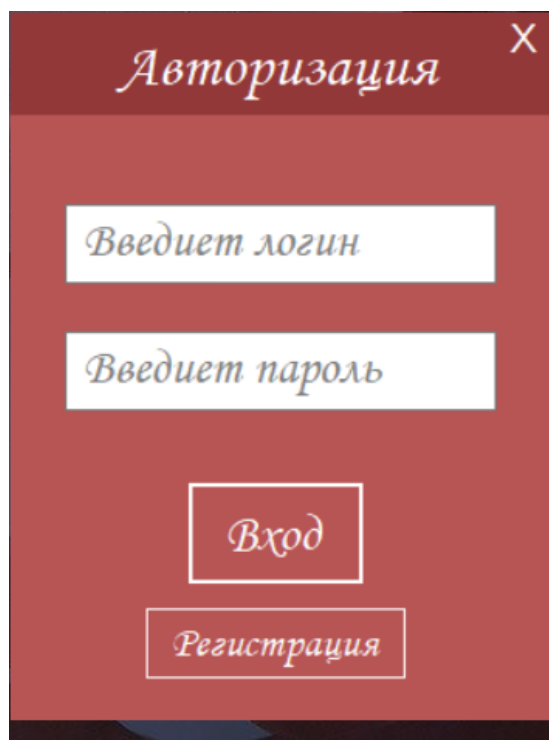


Рисунок 19 — Форма загрузки програми

Після прогрузки програми автоматично відкривається форма логіну де користувач повинен записати в поля свій логін та пароль. Форма логіну представлена на Рисунку 20.



Авторизация X

Введиет логин

Введиет пароль

Вход

Регистрация

Рисунок 20 — Форма логіну

Якщо користувач хоче сам себе реєструвати тоді він повинен натиснути на кнопку реєстрації після чого відкриється форма реєстрації користувача (Див. Рисунок 21). Логін та пароль може створити кожен однак при переходу до головної форми цей користувач буде мати статус #користувач і більша частина всіх можливостей програми буде заблокована.

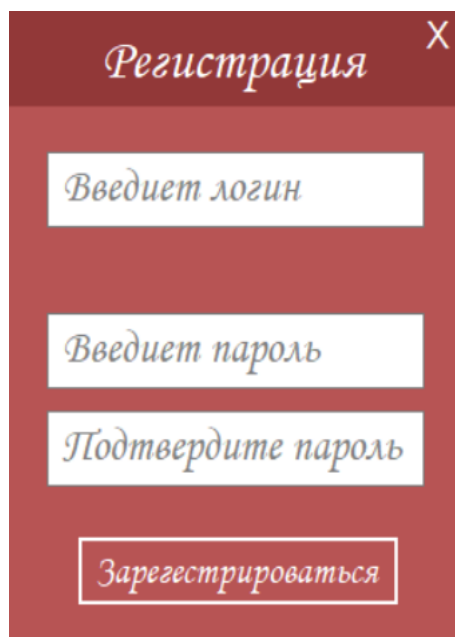
The image shows a registration form with a dark red background. At the top, the word "Регистрация" is written in a light-colored, stylized font, followed by a small white 'X' icon in the top right corner. Below the title, there are four white rectangular input fields stacked vertically. The first field contains the text "Введиет логин", the second "Введиет пароль", the third "Подтвердите пароль", and the fourth is a button labeled "Зарегистрироваться".

Рисунок 21 — Форма реєстрації користувача

Якщо користувач випадково натиснув на реєстрацію то щоб вийти з цієї форми необхідно натиснути на білий хрестик, який росташован у правому верхньому кутку. Після цього ця форма автоматично закриється та відкриється форма логіну.

Після коректного заповнення даних користувача при натисканні на кнопку входу з'явиться вікно з повідомленням, де повідомляється який статус має цей користувач (Див. Рисунок 22). Усього статусів 3: користувач, адміністратор та співробітник. У користувача більша частина всіх можливостей заблокована, а у співробітника навпаки, але є й те, що також заблоковано (наприклад перехід до сторінки phpMyAdmin).

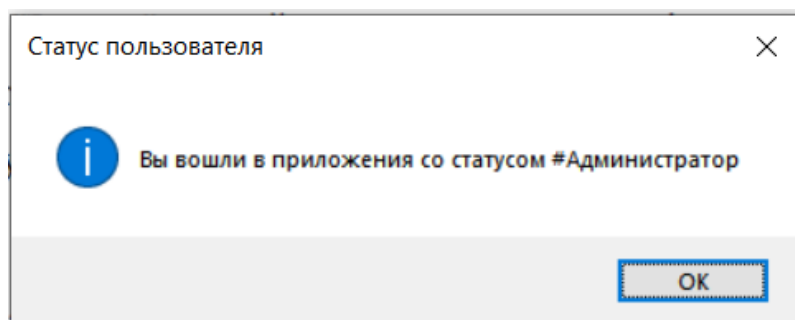


Рисунок 22 — Вікно повідомлення

Після входу до головної форми відкривається головна форма на котрій є деяке меню. Воно має такі пункти:

- "Файл"- даний пункт меню має два підпункти (Див. Рисунок 23) — Вихід та RHP.MyAdmin. Натиснув вихід користувач вийде з програми, а натиснув на RHP.MyAdmin відкриється сторінка браузеру з самою базою даних.

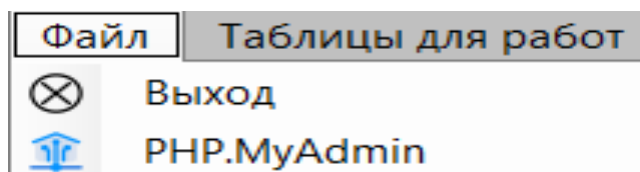


Рисунок 23 – Пункт меню файл

- "Таблицы для работ" – даний пункт меню має два підпункти – Заказы та Работа по заказам, при виборі яких користувач може перейти до форми просмотра, редагування та фільтрації (залежить від типу користувача; Див. Додаток А) даних (Див Рисунок 24).

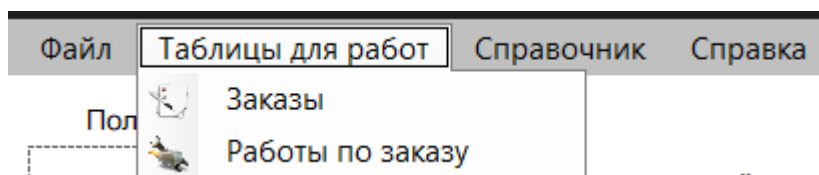


Рисунок 24 – Таблицы

– "Справочник" – даний пункт меню має чотири підпункти, деякі з яких мають тех підпункти — "Клиенты", "Менеджеры", "Специалисты" та "Доп. Информация" (Див. Рисунок 25,26,27). "Доп информация" мають три підпункти: "Материалы", "Услуги", "Статус заказов"; Підпункт "Специалисты" має один підпункт: "Специализация". При натисканні на один із цих підпунктів буде відкриватися нова форма з таблицею для редагування та фільтрації даних.

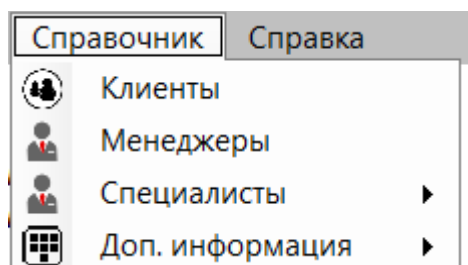


Рисунок 25 — "Справочник"



Рисунок 26 — Підпункт "Специалисты"



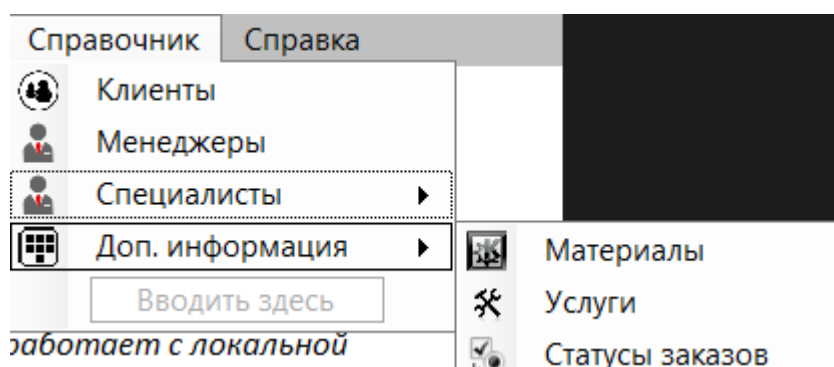


Рисунок 27— Подпункт “Доп. информация”

- "Справка" – при виборі даного пункту відкривається форма с інформацією о компанії (Див. Рисунок 28).

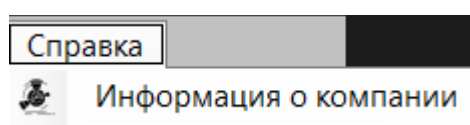


Рисунок 28 — Справка

Для редагування таблиць необхідно виділити запис та клікнув на пункт “Работа с данными” вибрати необхідний пункт (Рисунок 29).

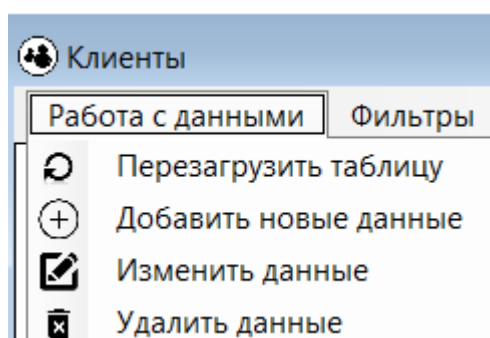


Рисунок 29 — Пункт “Работа с данными”

Для сортування даних по “зростанню” або “спаданню” необхідно клікнути на необхідний стовбчик (для всіх таблиць).

Також на деяких формах, а саме “Заказы”, “Клиенты” та

“Работі\_по\_заказам” є фільтри для пошуку деяких даних.

На формі “Клиенты” користувач може шукати пусті значення або заповнені. Для очищення фільтрів в пункті фільтрів є підпункт “Очистить фильтры” (Див. Рисунок 30, 31).

Фильтры	Вводитъ здесъ	label1
<input type="radio"/> Поиск пустых значений		Номер паспорта
<input checked="" type="checkbox"/> Поиск заполненный значений		ИНН
<input type="radio"/> Очистить фильтры		<input type="text" value="Вводитъ здесъ"/>

Рисунок 30 — Фільтри на формі клієнти

Фильтры	Вводитъ здесъ	label1
<input type="radio"/> Поиск пустых значений		
<input checked="" type="checkbox"/> Поиск заполненный значений		ФИО Место проживания Номер паспорта ИНН Номер телефона Реквизиты Логин Пароль
<input type="radio"/> Очистить фильтры	<input type="text" value="Вводитъ здесъ"/>	

Рисунок 31 — Фільтри на формі клієнти

На формі “Заказы” користувач може шукати пусті значення або пошук по статусу заказів. Для очищення фільтрів в пункті фільтрів є підпункт “Очистить фильтры” (Див. Рисунок 32, 33).

Фильтры	Отчёты	Вводитъ здесъ
<input type="radio"/> Поиск пустых значений		Дата сдачи заказа
<input type="radio"/> Статус заказа		<input type="text" value="Вводитъ здесъ"/>
<input type="radio"/> Очистить фильтры		

Рисунок 32 — Фільтри на формі закази

Фильтры	Отчёты	Вводить здесь
<input type="radio"/> Поиск пустых значений <input checked="" type="radio"/> Статус заказа <input type="radio"/> Очистить фильтры <input type="button" value="Вводить здесь"/>		Зарегистрировано В процессе Завершено

Рисунок 33 — Фільтри на формі закази

На формі “Работы по заказам” користувач може шукати роботи виконання певним рбїтником або пошук по послугі. Для очищення фільтрів в пункті фільтрів є підпункт “Очистить фильтры” (Див. Рисунок 34, 35).

Фильтры	Запросы	Отчёт по данной таблице	Вводить здесь
<input type="radio"/> Поиск по услуге <input type="radio"/> Поиск по специалисту <input type="radio"/> Очистить фильтры <input type="button" value="Вводить здесь"/>			1-Подключение телефония 2-Подключение ТВ 3-Установка ТКД 4-Подключение ТКД 5-Монтаж и запуск ТКД 6-Техпроблема

Рисунок 34 — Фільтри на формі закази

Фильтры	Запросы	Отчёт по данной таблице	Вводить здесь
<input type="radio"/> Поиск по услуге <input checked="" type="radio"/> Поиск по специалисту <input type="radio"/> Очистить фильтры <input type="button" value="Вводить здесь"/>			1-Акимов Иван Алексеевич 2-Кокоткин Роман Николаевич 3-Грандомаев Сергей Иванович

Рисунок 35 — Фільтри на формі закази

Також на цих трьох формах можна робити звіти клікнув на пункт “| Отчёт по данной таблице”. Натиснувши на цей пункт автоматично відчиняється браузер Google Chrome з відкритою вкладкою для створення необхідного звіту (Див. Рисунок 12).

## ВИСНОВКИ

У процесі розробки курсового проекту була створена база даних: «АИС для учета работы кабельного телевидения (предоставление услуг населению)». Програма була розроблена на основі базового алгоритму роботи з базою даних. Дана робота виконана в середовищі візуального програмування Visual Studio 2019.

У ході роботи були закріплені знання з дисциплін: «Інструментальні засоби візуального програмування», «Бази даних», «Людино-машинний інтерфейс», також були закріплені знання з дисципліни «Конструювання програмного забезпечення» при оформленні посібника користувача.

Отже, розроблена база даних може бути використана компаніями, та її можна вдосконалювати розширенням за допомогою додавання нових даних та коректування в залежності від потреб, удосконалювання пошуку і фільтрації даних. Створення конструктору запитів та конструктору звітів. Обов'язково необхідно розробити функцію резервного копіювання даних, а також відображення прогресса у виді діаграм.

## **ПЕРЕЛІК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ**

- 1) <https://docs.microsoft.com/ru-ru/sql/?view=sql-server-ver15>
- 2) [uk.wikipedia.org/wiki/](https://uk.wikipedia.org/wiki/)
- 3) <https://metanit.com/sharp/>

## ДОДАТОК А

	#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
<input type="checkbox"/>	1	ID_материала 🗑️	int(16)			Нет	Нем		AUTO_INCREMENT
<input type="checkbox"/>	2	Наименование	varchar(255)	utf8_general_ci		Нет	Нем		
<input type="checkbox"/>	3	Единица_измерения	varchar(16)	utf8_general_ci		Нет	Нем		
<input type="checkbox"/>	4	Количество	int(16)			Нет	Нем		
<input type="checkbox"/>	5	Цена	int(16)			Нет	Нем		
<input type="checkbox"/>	6	Валюта	varchar(16)	utf8_general_ci		Нет	Грн.		

Рисунок 36 — Конструктор таблиці матеріалів

	#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
<input type="checkbox"/>	1	ID_менеджера 🗑️	int(16)			Нет	Нем		AUTO_INCREMENT
<input type="checkbox"/>	2	ФИО	varchar(255)	utf8_general_ci		Нет	Нем		
<input type="checkbox"/>	3	Логин	varchar(16)	utf8_general_ci		Нет	Нем		
<input type="checkbox"/>	4	Пароль	varchar(16)	utf8_general_ci		Нет	Нем		

Рисунок 37 — Конструктор таблиці менеджерів

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
1	ID_клиента 	int(16)			Нет	Нет		AUTO_INCREMENT
2	ФИО	varchar(255)	utf8_general_ci		Нет	Нет		
3	Место_проживания	varchar(255)	utf8_general_ci		Нет	Нет		
4	Номер_паспорта	varchar(255)	utf8_general_ci		Да	NULL		
5	ИНН	varchar(255)	utf8_general_ci		Да	NULL		
6	Номер_телефона	varchar(255)	utf8_general_ci		Нет	Нет		
7	Реквизиты	varchar(255)	utf8_general_ci		Нет	Нет		
8	Логин	varchar(16)	utf8_general_ci		Нет	Нет		
9	Пароль	varchar(16)	utf8_general_ci		Нет	Нет		

Рисунок 38 — Конструктор таблиці клієнтів

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
1	ID 	int(16)			Нет	Нет		AUTO_INCREMENT
2	Логин	varchar(16)	utf8_general_ci		Нет	Нет		
3	Пароль	varchar(16)	utf8_general_ci		Нет	Нет		

Рисунок 39 — Конструктор таблиці користувачів

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
1	ID_заказа 	int(16)			Нет	Нет		AUTO_INCREMENT
2	Дата_принятия_заказ	date			Нет	Нет		
3	Дата_сдачи_заказа	date			Да	NULL		
4	id_клиента 	int(16)			Нет	Нет		
5	id_менеджера 	int(16)			Нет	Нет		
6	id_специалиста 	int(16)			Нет	Нет		
7	id_статуса 	int(16)			Нет	Нет		

Рисунок 40 — Конструктор таблиці заказів

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
1	ID_работ 🏢	int(16)			Нет	Нет		AUTO_INCREMENT
2	id_заказа 🏢	int(16)			Нет	Нет		
3	id_услуги 🏢	int(16)			Нет	Нет		
4	Количество_материалов	int(16)			Нет	Нет		
5	id_материала 🏢	int(16)			Нет	Нет		
6	id_специалиста 🏢	int(16)			Нет	Нет		
7	Дата	date			Да	NULL		
8	Цена	int(16)			Нет	Нет		
9	Валюта	varchar(16)	utf8_general_ci		Нет	Грн.		

Рисунок 41 — Конструктор таблиці работ по заказам

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
1	ID_специализации 🏢	int(16)			Нет	Нет		AUTO_INCREMENT
2	Специализация	varchar(255)	utf8_general_ci		Нет	Нет		

Рисунок 42 — Конструктор таблиці спеціалізації

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
1	ID_специалиста 🏢	int(16)			Нет	Нет		AUTO_INCREMENT
2	ФИО	varchar(255)	utf8_general_ci		Нет	Нет		
3	id_специализации 🏢	int(16)			Нет	Нет		
4	Логин	varchar(16)	utf8_general_ci		Нет	Нет		
5	Пароль	varchar(16)	utf8_general_ci		Нет	Нет		

Рисунок 43 — Конструктор таблиці спеціалістів

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
1	ID_услуги 🏢	int(16)			Нет	Нет		AUTO_INCREMENT
2	Наименование	varchar(255)	utf8_general_ci		Нет	Нет		
3	Ценовая политика	int(16)			Нет	Нет		
4	Валюта	varchar(16)	utf8_general_ci		Нет	Грн.		

Рисунок 43 — Конструктор таблиці услуг

#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
1	ID_статуса 🏢	int(16)			Нет	Нет		AUTO_INCREMENT
2	Статус	varchar(255)	utf8_general_ci		Нет	Нет		

Рисунок 44 — Конструктор таблиці статусів заказів



Свойства ограничения	Столбец ⓘ	Ограничение внешнего ключа (INNODB)		
		База данных	Таблица	Столбец
специалисты_ibfk_1 ON DELETE RESTRICT ON UPDATE CASCADE	id_специализации <a href="#">+ Добавить столбец</a>	bd_tv_services	специализация	ID_специализации

Рисунок 45 — Зв'язок в таблиці спеціалісти

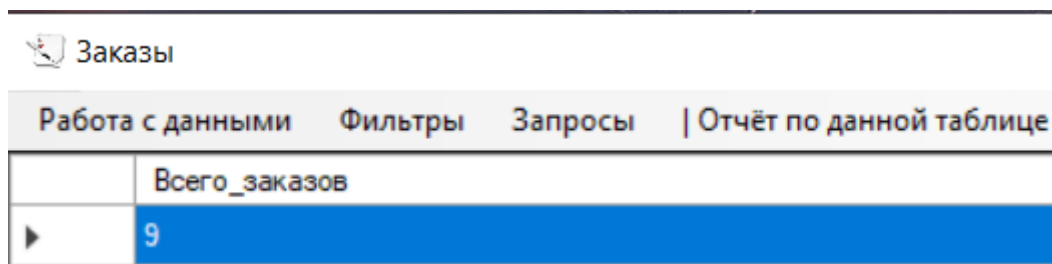
Свойства ограничения	Столбец ⓘ	Ограничение внешнего ключа (INNODB)		
		База данных	Таблица	Столбец
работы_по_заказу_ibfk_1 ON DELETE RESTRICT ON UPDATE CASCADE	id_заказа <a href="#">+ Добавить столбец</a>	bd_tv_services	заказ	ID_заказа
работы_по_заказу_ibfk_2 ON DELETE RESTRICT ON UPDATE CASCADE	id_специалиста <a href="#">+ Добавить столбец</a>	bd_tv_services	специалисты	ID_специалиста
работы_по_заказу_ibfk_3 ON DELETE RESTRICT ON UPDATE CASCADE	id_услуги <a href="#">+ Добавить столбец</a>	bd_tv_services	услуги	ID_услуги
работы_по_заказу_ibfk_4 ON DELETE RESTRICT ON UPDATE CASCADE	id_материала <a href="#">+ Добавить столбец</a>	bd_tv_services	материалы	ID_материала

Рисунок 46 — Зв'язок в таблиці роботи по заказам

Свойства ограничения	Столбец ⓘ	Ограничение внешнего ключа (INNODB)		
		База данных	Таблица	Столбец
заказ_ibfk_1 ON DELETE RESTRICT ON UPDATE CASCADE	id_менеджера <a href="#">+ Добавить столбец</a>	bd_tv_services	менеджеры	ID_менеджера
заказ_ibfk_2 ON DELETE RESTRICT ON UPDATE CASCADE	id_клиента <a href="#">+ Добавить столбец</a>	bd_tv_services	клиенты	ID_клиента
заказ_ibfk_3 ON DELETE RESTRICT ON UPDATE CASCADE	id_специалиста <a href="#">+ Добавить столбец</a>	bd_tv_services	специалисты	ID_специалиста
заказ_ibfk_4 ON DELETE RESTRICT ON UPDATE CASCADE	id_статуса <a href="#">+ Добавить столбец</a>	bd_tv_services	статус_заказа	ID_статуса

Рисунок 47 — Зв'язок в таблиці заклази

## ДОДАТОК Б



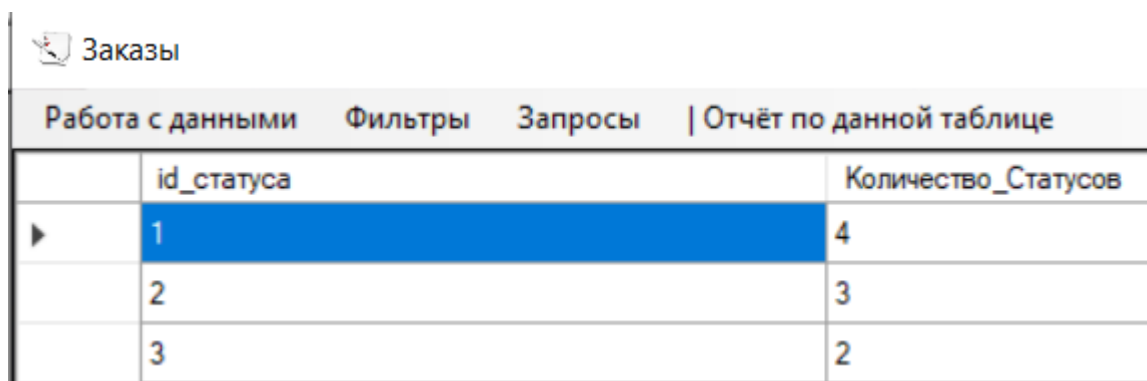
Заказы	
Работа с данными   Фильтры   Запросы     Отчёт по данной таблице	
	Всего_заказов
▶	9

Рисунок 48— Результат работы запроса “Количество заказов”

```
private void количествоToolStripMenuItem_Click(object sender, EventArgs e)
{
    db = new DataBase();
    db.openConnection();
    adapted = new MySqlDataAdapter("SELECT COUNT(ID_заказа) AS Всего_заказов FROM
`заказ`", db.getConnection());
    table = new DataTable();

    adapted.Fill(table);
    dataGridView1.DataSource = table;
    db.closeConection();
}
```

Рисунок 49— Код запроса “Количество заказов”



Заказы	
Работа с данными   Фильтры   Запросы     Отчёт по данной таблице	
	id_статуса
▶	1
	2
	3

Рисунок 50— Результат работы запроса “Подсчет заказов по статусам”

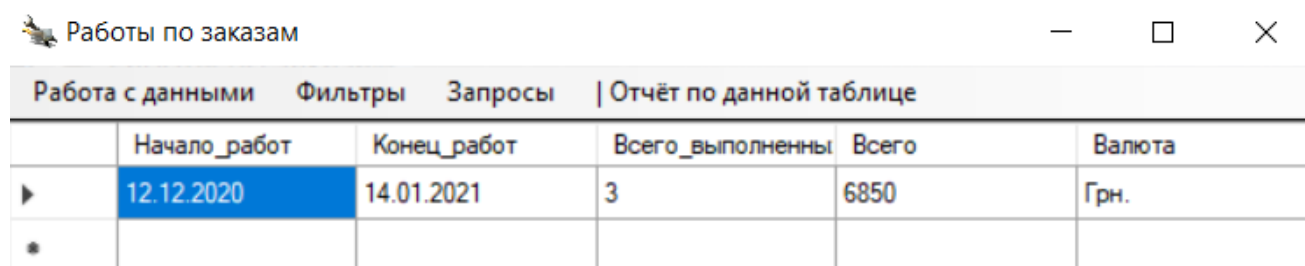
```

private void подсчетЗаказавоПоСтатусамToolStripMenuItem_Click(object sender, EventArgs e)
{
    db = new DataBase();
    db.openConnection();
    adapted = new MySqlDataAdapter("SELECT id_статуса, COUNT(id_статуса) AS
Количество_Статусов FROM `заказ` GROUP BY id_статуса", db.getConnection());
    table = new DataTable();

    adapted.Fill(table);
    dataGridView1.DataSource = table;
    db.closeConnection();
}

```

Рисунок 51— Код запити “Подсчет заказов по статусам”



	Начало_работ	Конец_работ	Всего_выполненны	Всего	Валюта
▶	12.12.2020	14.01.2021	3	6850	Грн.
•					

Рисунок 52— Результат работы запити запити “Общая сумма выполненных работ”

```

private void общаяСуммаВыполненныхРаботToolStripMenuItem_Click(object sender, EventArgs e)
{
    table = new DataTable();
    db = new DataBase();

    table.Clear();
    db.openConnection();

    adapted = new MySqlDataAdapter("SELECT MIN(`работы_по_заказу`.`Дата`) AS
Начало_работ, MAX(`работы_по_заказу`.`Дата`) AS Конец_работ,
COUNT(`работы_по_заказу`.`ID_работ`) AS Всего_выполненных_работ,
SUM(`работы_по_заказу`.`Цена`) AS Всего, `работы_по_заказу`.`Валюта` FROM `работы_по_заказу`
WHERE `работы_по_заказу`.`Дата` IS NOT NULL", db.getConnection());
    adapted.Fill(table);
    dataGridView1.DataSource = table;

    db.closeConnection();
}

```

Рисунок 53— Код запити запити “Общая сумма выполненных работ”

## ДОДАТОК С

### Program.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace WindowsFormsApp1
{
    static class Program
    {
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new FormBar());
        }
    }
}
```

### DataBase.cs

```
using System;
using MySql.Data.MySqlClient;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace WindowsFormsApp1
{
    class DataBase
    {
        MySqlConnection connection = new MySqlConnection("server=localhost; port=3306;
        User=root; Password=root; database=bd_tv_services");

        public void openConection()
        {
            if (connection.State == System.Data.ConnectionState.Closed)
                connection.Open();
        }

        public void closeConection()
        {
            if (connection.State == System.Data.ConnectionState.Open)
                connection.Close();
        }

        public MySqlConnection getConnection()
        {
            return connection;
        }
    }
}
```

## LoginForm.cs

```

using MySql.Data.MySqlClient;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace WindowsFormsApp1
{
    public partial class LoginForm : Form
    {
        public LoginForm()
        {
            InitializeComponent();

            this.StartPosition = FormStartPosition.CenterScreen;

            loginField.ForeColor = Color.Gray;
            passField.ForeColor = Color.Gray;
            passField.Multiline = true;
            loginField.Text = "Введиет логин";
            passField.Text = "Введиет пароль";

            this.passField.Size = new Size(this.passField.Size.Width, 42);
            this.passField.AutoSize = false;

            this.loginField.Size = new Size(this.loginField.Size.Width, 42);
            this.loginField.AutoSize = false;
        }

        private void OutClickButton_Click(object sender, EventArgs e)
        {
            Application.Exit();
        }

        private void OutClickButton_MouseEnter(object sender, EventArgs e)
        {
            OutClickButton.Font = new Font("OutClickButton", 20, FontStyle.Regular);
        }

        private void OutClickButton_MouseLeave(object sender, EventArgs e)
        {
            OutClickButton.Font = new Font("OutClickButton", 15, FontStyle.Regular);
        }

        Point LastPoint;
        private void panel2_MouseMove(object sender, MouseEventArgs e)
        {
            if(e.Button == MouseButtons.Left)
            {
                this.Left += e.X - LastPoint.X;
                this.Top += e.Y - LastPoint.Y;
            }
        }

        private void panel2_MouseDown(object sender, MouseEventArgs e)

```

```

{
    LastPoint = new Point(e.X, e.Y);
}

private void label1_MouseDown(object sender, MouseEventArgs e)
{
    LastPoint = new Point(e.X, e.Y);
}

private void buttonlogin_Click(object sender, EventArgs e)
{
    String loginUser = loginField.Text;
    String passUser = passField.Text;

    DataBase db = new DataBase();
    DataTable table = new DataTable();
    MySqlDataAdapter adapted = new MySqlDataAdapter();

    MySqlCommand command = new MySqlCommand("SELECT * FROM `пользователи` WHERE
`Логин` = @uL AND `Пароль` = @uP", db.getConnection());
    command.Parameters.Add("@uL", MySqlDbType.VarChar).Value = loginUser;
    command.Parameters.Add("@uP", MySqlDbType.VarChar).Value = passUser;

    adapted.SelectCommand = command;
    adapted.Fill(table);
    if (table.Rows.Count > 0)
    {
        this.Hide();
        MainForm mainForm = new MainForm();
        mainForm.User_save = this.loginField.Text;
        mainForm.Show();
    }
    else
        MessageBox.Show("Ошибка авторизации");
}

private void buttonregister_Click(object sender, EventArgs e)
{
    this.Hide();
    RegistForm newForm = new RegistForm();
    newForm.Show();
}

private void loginField_Enter(object sender, EventArgs e)
{
    if (loginField.Text == "Вводит логин")
    {
        loginField.ForeColor = Color.Black;
        loginField.Text = "";
    }
}

private void loginField_Leave(object sender, EventArgs e)
{
    if (loginField.Text == "")
    {
        loginField.Text = "Вводит логин";
        loginField.ForeColor = Color.Gray;
    }
}

private void passField_Leave(object sender, EventArgs e)
{
    if (passField.Text == "")

```

```

        {
            passField.Text = "Введиет пароль";
            passField.Multiline = true;
            passField.ForeColor = Color.Gray;
        }
    }

    private void passField_Enter(object sender, EventArgs e)
    {
        if (passField.Text == "Введиет пароль")
        {
            passField.Text = "";
            passField.Multiline = false;
            passField.ForeColor = Color.Black;
        }
    }
}

```

## RegistForm.cs

```

using MySql.Data.MySqlClient;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace WindowsFormsApp1
{
    public partial class RegistForm : Form
    {
        public RegistForm()
        {
            InitializeComponent();

            this.StartPosition = FormStartPosition.CenterScreen;

            loginRegistField.Text = "Введиет логин";
            passRegistField.Text = "Введиет пароль";
            passRegistFieldReturn.Text = "Подтвердите пароль";
            passRegistField.Multiline = true;
            passRegistFieldReturn.Multiline = true;
            loginRegistField.ForeColor = Color.Gray;
            passRegistField.ForeColor = Color.Gray;
            passRegistFieldReturn.ForeColor = Color.Gray;

            this.passRegistField.Size = new Size(this.passRegistField.Size.Width, 42);
            this.passRegistField.AutoSize = false;

            this.passRegistFieldReturn.Size = new Size(this.passRegistField.Size.Width, 42);
            this.passRegistFieldReturn.AutoSize = false;

            this.loginRegistField.Size = new Size(this.loginRegistField.Size.Width, 42);
            this.loginRegistField.AutoSize = false;
        }

        private void OutClickButton_Click(object sender, EventArgs e)
        {
            this.Hide();
            LoginForm loginForm = new LoginForm();

```

```

        loginForm.Show();
    }

    private void OutClickButton_MouseEnter_1(object sender, EventArgs e)
    {
        OutClickButton.Font = new Font("OutClickButton", 20, FontStyle.Regular);
    }

    private void OutClickButton_MouseLeave_1(object sender, EventArgs e)
    {
        OutClickButton.Font = new Font("OutClickButton", 15, FontStyle.Regular);
    }

    Point LastPoint;
    private void panel2_MouseMove(object sender, MouseEventArgs e)
    {
        if (e.Button == MouseButtons.Left)
        {
            this.Left += e.X - LastPoint.X;
            this.Top += e.Y - LastPoint.Y;
        }
    }

    private void panel2_MouseDown(object sender, MouseEventArgs e)
    {
        LastPoint = new Point(e.X, e.Y);
    }

    private void label1_MouseDown(object sender, MouseEventArgs e)
    {
        LastPoint = new Point(e.X, e.Y);
    }

    private void loginRegistField_Enter(object sender, EventArgs e)
    {
        if (loginRegistField.Text == "Введиет лoгин")
        {
            loginRegistField.Text = "";
            loginRegistField.ForeColor = Color.Black;
        }
    }

    private void loginRegistField_Leave(object sender, EventArgs e)
    {
        if (loginRegistField.Text == "")
        {
            loginRegistField.Text = "Введиет лoгин";
            loginRegistField.ForeColor = Color.Gray;
        }
    }

    private void passRegistField_Enter(object sender, EventArgs e)
    {
        if (passRegistField.Text == "Введиет пaрoль")
        {
            passRegistField.Text = "";
            passRegistField.Multiline = false;
            passRegistField.ForeColor = Color.Black;
        }
    }

    private void passRegistField_Leave(object sender, EventArgs e)
    {
        if (passRegistField.Text == "")
        {
            passRegistField.Text = "Введиет пaрoль";
        }
    }

```



```

        passRegistField.Multiline = true;
        passRegistField.ForeColor = Color.Gray;
    }
}

private void passRegistFieldReturn_Leave(object sender, EventArgs e)
{
    if (passRegistFieldReturn.Text == "")
    {
        passRegistFieldReturn.Text = "Подтвердите пароль";
        passRegistFieldReturn.Multiline = true;
        passRegistFieldReturn.ForeColor = Color.Gray;
    }
}

private void passRegistFieldReturn_Enter(object sender, EventArgs e)
{
    if (passRegistFieldReturn.Text == "Подтвердите пароль")
    {
        passRegistFieldReturn.Text = "";
        passRegistFieldReturn.Multiline = false;
        passRegistFieldReturn.ForeColor = Color.Black;
    }
}

private void buttonregister_Click(object sender, EventArgs e)
{
    if (loginRegistField.Text == "Введиет логин")
    {
        MessageBox.Show("Не корректный ввод данных");
        return;
    }

    if (passRegistFieldReturn.Text != passRegistField.Text)
    {
        MessageBox.Show("Пароли не совпадают");
        return;
    }

    if (isUsersExists())
        return;

    String loginRegistUser = loginRegistField.Text;
    String passRegistUser = passRegistField.Text;

    DataBase db = new DataBase();
    MySqlCommand command = new MySqlCommand("INSERT INTO `пользователи` (`Логин`, `Пароль`) VALUES (@login, @pass)", db.getConnection());
    command.Parameters.Add("@login", MySqlDbType.VarChar).Value = loginRegistField.Text;
    command.Parameters.Add("@pass", MySqlDbType.VarChar).Value = passRegistField.Text;

    db.openConnection();

    if (command.ExecuteNonQuery() == 1)
        MessageBox.Show("Аккаунт был создан");
    else
        MessageBox.Show("ошибка создания аккаунта");

    db.closeConection();

    this.Hide();
    LoginForm loginForm = new LoginForm();
    loginForm.Show();
}

```

```

    }

    public Boolean isUsersExists()
    {
        DataBase db = new DataBase();
        DataTable table = new DataTable();
        MySqlDataAdapter adapted = new MySqlDataAdapter();

        MySqlCommand command = new MySqlCommand("SELECT * FROM `пользователи` WHERE
`Логин` = @uL", db.getConnection());
        command.Parameters.Add("@uL", MySqlDbType.VarChar).Value = loginRegistField.Text;

        adapted.SelectCommand = command;
        adapted.Fill(table);
        if (table.Rows.Count > 0)
        {
            MessageBox.Show("Данный логин уже существует");
            return true;
        }
        else
        {
            return false;
        }
    }
}
}

```

## mainForm.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace WindowsFormsApp1
{
    public partial class mainForm : Form
    {
        public mainForm()
        {
            InitializeComponent();
            this.StartPosition = FormStartPosition.CenterScreen;
        }

        public string User_save
        {
            get { return label_user.Text; }
            set { label_user.Text = value; }
        }

        private void информацияОКомпанииToolStripMenuItem_Click(object sender, EventArgs e)
        {
            FormInfo formInfo = new FormInfo();
            formInfo.Show();
        }

        private void выходToolStripMenuItem_Click(object sender, EventArgs e)
        {

```

```

        if (MessageBox.Show("Подтвердить выход из приложения?", "Выход из приложения",
        MessageBoxButtons.YesNo) == System.Windows.Forms.DialogResult.Yes)
        {
            Application.Exit();
        }
    }

    private void пользователиToolStripMenuItem_Click(object sender, EventArgs e)
    {
        clientsForm clientsForm = new clientsForm();
        clientsForm.User_save = label_user.Text;
        clientsForm.Show();
    }

    private void менеджерыToolStripMenuItem_Click(object sender, EventArgs e)
    {
        managersForm managersForm = new managersForm();
        managersForm.User_save = label_user.Text;
        managersForm.Show();
    }

    private void рабочиеToolStripMenuItem_Click(object sender, EventArgs e)
    {
        workersForm workersForm = new workersForm();
        workersForm.User_save = label_user.Text;
        workersForm.Show();
    }

    private void специализацияToolStripMenuItem_Click(object sender, EventArgs e)
    {
        workers_specz_Form workers_specz_Form = new workers_specz_Form();
        workers_specz_Form.User_save = label_user.Text;
        workers_specz_Form.Show();
    }

    private void материалыToolStripMenuItem1_Click(object sender, EventArgs e)
    {
        materialForm materialForm = new materialForm();
        materialForm.Show();
    }

    private void услугиToolStripMenuItem_Click(object sender, EventArgs e)
    {
        servicesForm servicesForm = new servicesForm();
        servicesForm.Show();
    }

    private void заказыToolStripMenuItem_Click(object sender, EventArgs e)
    {
        ordersForm ordersForm = new ordersForm();
        ordersForm.User_save = label_user.Text;
        ordersForm.Show();
    }

    private void работыПоЗаказуToolStripMenuItem_Click(object sender, EventArgs e)
    {
        works_of_ordersForm works_of_ordersForm = new works_of_ordersForm();
        works_of_ordersForm.Show();
    }

    private void статусыЗаказовToolStripMenuItem_Click(object sender, EventArgs e)
    {
        statusForm statusForm = new statusForm();
        statusForm.Show();
    }

```

```

Point LastPoint;
private void mainForm_MouseDown(object sender, MouseEventArgs e)
{
    LastPoint = new Point(e.X, e.Y);
}

private void mainForm_MouseMove(object sender, MouseEventArgs e)
{
    if (e.Button == MouseButtons.Left)
    {
        this.Left += e.X - LastPoint.X;
        this.Top += e.Y - LastPoint.Y;
    }
}

private void pictureBox1_MouseMove(object sender, MouseEventArgs e)
{
    if (e.Button == MouseButtons.Left)
    {
        this.Left += e.X - LastPoint.X;
        this.Top += e.Y - LastPoint.Y;
    }
}

private void pictureBox1_MouseDown(object sender, MouseEventArgs e)
{
    LastPoint = new Point(e.X, e.Y);
}

private void label1_MouseDown(object sender, MouseEventArgs e)
{
    LastPoint = new Point(e.X, e.Y);
}

private void label1_MouseMove(object sender, MouseEventArgs e)
{
    if (e.Button == MouseButtons.Left)
    {
        this.Left += e.X - LastPoint.X;
        this.Top += e.Y - LastPoint.Y;
    }
}

private void menuStrip1_MouseDown(object sender, MouseEventArgs e)
{
    LastPoint = new Point(e.X, e.Y);
}

private void menuStrip1_MouseMove(object sender, MouseEventArgs e)
{
    if (e.Button == MouseButtons.Left)
    {
        this.Left += e.X - LastPoint.X;
        this.Top += e.Y - LastPoint.Y;
    }
}

public void Count()
{
    string str = User_save;
    for (int i = 0; i < str.Length; i++)
        if (str[0] == 'm' && str[1] == 'a' && str[2] == 'n' && str[3] == 'a' &&
str[4] == 'g' && str[5] == 'e' && str[6] == 'r' || str[0] == 'w' && str[1] == 'o' && str[2]
== 'r' && str[3] == 'k' && str[4] == 'i' && str[5] == 'n' && str[6] == 'g')

```

```

    {
        MessageBox.Show(
            "Вы вошли в приложения со статусом #Сотрудник",
            "Статус пользователя",
            MessageBoxButtons.OK,
            MessageBoxIcon.Information);
        break;
    }
    else if (label_user.Text == "admin" || label_user.Text == "1")
    {
        MessageBox.Show(
            "Вы вошли в приложения со статусом #Администратор",
            "Статус пользователя",
            MessageBoxButtons.OK,
            MessageBoxIcon.Information);

        break;
    }
    else
    {
        MessageBox.Show(
            "Вы вошли в приложения со статусом #Клиент",
            "Статус пользователя",
            MessageBoxButtons.OK,
            MessageBoxIcon.Information);
        дополнительнаяИнформацияToolStripMenuItem.Visible = false;
        таблицыДляРаботыToolStripMenuItem.Visible = false;
        break;
    }
}

private void mainForm_Load(object sender, EventArgs e)
{
    Count();
    if (label_user.Text == "1" || label_user.Text == "admin")
    {
    }
    else
        создатьОтчетВручнуюToolStripMenuItem.Visible = false;
}

private void создатьОтчетВручнуюToolStripMenuItem_Click(object sender, EventArgs e)
{
    WebBrowser web = new WebBrowser();
    System.Diagnostics.Process.Start(@"chrome.exe",
"http://127.0.0.1/openserver/phpmyadmin/db_structure.php?server=1&db=bd_tv_services");
}
}
}

```

## orderForm.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;
using Excel = Microsoft.Office.Interop.Excel;

```

```

namespace WindowsFormsApp1
{
    public partial class ordersForm : Form
    {
        private DataBase db = null;
        private DataTable table = null;
        private MySqlDataAdapter adapted = null;
        private MySqlCommand command_insert = null;
        private MySqlCommand command = null;
        private MySqlCommand command_update = null;
        private MySqlCommand command_delete = null;

        public ordersForm()
        {
            InitializeComponent();
        }

        public string User_save
        {
            get { return label1.Text; }
            set { label1.Text = value; }
        }
        public void Count()
        {
            string str = User_save;
            for(int i=0; i<str.Length; i++)
                if(str[0] == 'm' && str[1] == 'a' && str[2] == 'n' && str[3] == 'a' && str[4] ==
'g' && str[5] == 'e' && str[6] == 'r')
                {
                    break;
                }
            else if (label1.Text == "1" || label1.Text == "admin")
            {
                break;
            }
            else
            {
                добавитьНовыеДанныеToolStripMenuItem.Visible = false;
                изменитьДанныеToolStripMenuItem.Visible = false;
                удалитьДанныеToolStripMenuItem.Visible = false;

                отчетыToolStripMenuItem.Visible = false;
                dataGridView1.AllowUserToAddRows = false;
                dataGridView1.AllowUserToDeleteRows = false;
                dataGridView1.ReadOnly = true;
                break;
            }
        }
        private void ordersForm_Load(object sender, EventArgs e)
        {
            Count();

            db = new DataBase();
            db.openConnection();
            adapted = new MySqlDataAdapter("SELECT * FROM `заказ`", db.getConnection());
            table = new DataTable();

            adapted.Fill(table);
            dataGridView1.DataSource = table;
            db.closeConection();
        }

        private void перезагрузитьБазуДанныхToolStripMenuItem_Click(object sender, EventArgs e)
        {
            db = new DataBase();
            db.openConnection();
            adapted = new MySqlDataAdapter("SELECT * FROM `заказ`", db.getConnection());
            table = new DataTable();

```

```

        adapted.Fill(table);
        dataGridView1.DataSource = table;
        db.closeConection();
    }

    private void добавитьНовыеДанныеToolStripMenuItem_Click(object sender, EventArgs e)
    {
        db = new DataBase();

        if (dataGridView1.SelectedRows.Count != 1)
        {
            MessageBox.Show("Выберите одну строку", "Внимание!");
            return;
        }

        int index = dataGridView1.SelectedRows[0].Index;

        db.openConection();

        command_insert = new MySqlCommand("INSERT INTO `заказ` (`Дата_принятия_заказ`,
`Дата_сдачи_заказа`, `id_клиента`, `id_менеджера`, `id_специалиста`, `id_статуса`) VALUES (@ff1,
@ff2, @ff3, @ff4, @ff5, @ff6);", db.getConnection());
        command_insert.Parameters.Add("@ff1", MySqlDbType.Date).Value =
dataGridView1.Rows[index].Cells[1].Value;
        command_insert.Parameters.Add("@ff2", MySqlDbType.Date).Value =
dataGridView1.Rows[index].Cells[2].Value;
        command_insert.Parameters.Add("@ff3", MySqlDbType.Int16).Value =
dataGridView1.Rows[index].Cells[3].Value;
        command_insert.Parameters.Add("@ff4", MySqlDbType.Int16).Value =
dataGridView1.Rows[index].Cells[4].Value;
        command_insert.Parameters.Add("@ff5", MySqlDbType.Int16).Value =
dataGridView1.Rows[index].Cells[5].Value;
        command_insert.Parameters.Add("@ff6", MySqlDbType.Int16).Value =
dataGridView1.Rows[index].Cells[6].Value;

        try
        {
            if (command_insert.ExecuteNonQuery() == 1)
                MessageBox.Show("Запись была добавлена");
            else
                MessageBox.Show("Ошибка добавления данных");
        }
        catch
        {
            MessageBox.Show("Ошибка обработки данных");
            return;
        }
        finally
        {
            db.closeConection();
        }
    }

    private void изменитьДанныеToolStripMenuItem_Click(object sender, EventArgs e)
    {
        if (dataGridView1.SelectedRows.Count != 1)
        {
            MessageBox.Show("Выберите одну строку", "Внимание!");
            return;
        }

        int index = dataGridView1.SelectedRows[0].Index;

        db.openConection();

        command_update = new MySqlCommand("UPDATE `заказ` SET `Дата_принятия_заказ` = @f1,
`Дата_сдачи_заказа` = @f2, `id_клиента` = @f3, `id_менеджера` = @f4, `id_специалиста` = @f5,
`id_статуса` = @f6 WHERE `заказ`.`ID_заказа` = @f0", db.getConnection());
        command_update.Parameters.Add("@f1", MySqlDbType.Date).Value =
dataGridView1.Rows[index].Cells[1].Value;

```

```

        command_update.Parameters.Add("@f2", MySqlDbType.Date).Value =
dataGridView1.Rows[index].Cells[2].Value;
        command_update.Parameters.Add("@f3", MySqlDbType.Int16).Value =
dataGridView1.Rows[index].Cells[3].Value;
        command_update.Parameters.Add("@f4", MySqlDbType.Int16).Value =
dataGridView1.Rows[index].Cells[4].Value;
        command_update.Parameters.Add("@f5", MySqlDbType.Int16).Value =
dataGridView1.Rows[index].Cells[5].Value;
        command_update.Parameters.Add("@f6", MySqlDbType.Int16).Value =
dataGridView1.Rows[index].Cells[6].Value;
        command_update.Parameters.Add("@f0", MySqlDbType.Int16).Value =
dataGridView1.Rows[index].Cells[0].Value;

```

```

        try
        {
            if (command_update.ExecuteNonQuery() == 1)
                MessageBox.Show("Запись была обновлена");
            else
                MessageBox.Show("Ошибка обновления данных"); ;
        }
        catch
        {
            MessageBox.Show("Ошибка обработки данных");
            return;
        }
        finally
        {
            db.closeConection();
        }
    }

```

```

private void удалитьДанныеToolStripMenuItem_Click(object sender, EventArgs e)
{
    if (dataGridView1.SelectedRows.Count != 1)
    {
        MessageBox.Show("Выберите одну строку", "Внимание!");
        return;
    }

    int index = dataGridView1.SelectedRows[0].Index;

    db.openConection();

    command_delete = new MySqlCommand("DELETE FROM `заказ` WHERE `заказ`.`ID_заказа` =
    @f0", db.getConnection());
    command_delete.Parameters.Add("@f0", MySqlDbType.Int16).Value =
dataGridView1.Rows[index].Cells[0].Value;

```

```

        try
        {
            if (command_delete.ExecuteNonQuery() == 1)
                MessageBox.Show("Данные были удалены");
            else
                MessageBox.Show("Ошибка удаления данных");
        }
        catch
        {
            MessageBox.Show("Ошибка обработки данных");
            return;
        }
        finally
        {
            db.closeConection();
        }
    }
}

```

```

private void зарегистрированоToolStripMenuItem_Click(object sender, EventArgs e)
{
    table = new DataTable();
    db = new DataBase();

```



```

        table.Clear();
        db.openConnection();

        adapted = new MySqlDataAdapter("SELECT * FROM `заказ` WHERE `id_сратыса` = 1",
db.getConnection());
        adapted.Fill(table);
        dataGridView1.DataSource = table;

        db.closeConection();
    }

    private void вПроцессеToolStripMenuItem_Click(object sender, EventArgs e)
    {
        table = new DataTable();
        db = new DataBase();

        table.Clear();
        db.openConnection();

        adapted = new MySqlDataAdapter("SELECT * FROM `заказ` WHERE `id_сратыса` = 2",
db.getConnection());
        adapted.Fill(table);
        dataGridView1.DataSource = table;

        db.closeConection();
    }

    private void завершеноToolStripMenuItem_Click(object sender, EventArgs e)
    {
        table = new DataTable();
        db = new DataBase();

        table.Clear();
        db.openConnection();

        adapted = new MySqlDataAdapter("SELECT * FROM `заказ` WHERE `id_сратыса` = 3",
db.getConnection());
        adapted.Fill(table);
        dataGridView1.DataSource = table;

        db.closeConection();
    }

    private void датасдачизаказаToolStripMenuItem_Click(object sender, EventArgs e)
    {
        table = new DataTable();
        db = new DataBase();

        table.Clear();
        db.openConnection();

        adapted = new MySqlDataAdapter("SELECT * FROM `заказ` WHERE `Дата_сдачи_заказа` IS
NULL", db.getConnection());
        adapted.Fill(table);
        dataGridView1.DataSource = table;

        db.closeConection();
    }

    private void очиститьФильтрыToolStripMenuItem_Click(object sender, EventArgs e)
    {
        db = new DataBase();
        db.openConnection();
        adapted = new MySqlDataAdapter("SELECT * FROM `заказ`", db.getConnection());
        table = new DataTable();

        adapted.Fill(table);
        dataGridView1.DataSource = table;
        db.closeConection();
    }

```

```

    }

    void Save_All_report(DataGridView Save_data)
    {
        if (dataGridView1.SelectedRows.Count <= 0)
        {
            MessageBox.Show("Выберите одну строку", "Внимание!");
            return;
        }

        string part = System.IO.Directory.GetCurrentDirectory() + @"\\" +
"Report_All_data.xlsx";
        Excel.Application excelapp = new Excel.Application();
        Excel.Workbook workbook = excelapp.Workbooks.Add();
        Excel.Worksheet worksheet = workbook.ActiveSheet;
        for (int i = 1; i < Save_data.RowCount + 1; i++)
        {
            for (int j = 0; j < Save_data.ColumnCount + 1; j++)
            {
                worksheet.Rows[i].Columns[j] = Save_data.Rows[i - 1].Cells[j - 1].Value;
            }
        }
        excelapp.AlertBeforeOverwriting = false;
        workbook.SaveAs(part);
        excelapp.Quit();
    }

    private void количествоToolStripMenuItem_Click(object sender, EventArgs e)
    {
        db = new DataBase();
        db.openConnection();
        adapted = new MySqlDataAdapter("SELECT COUNT(ID_заказа) AS Всего_заказов FROM
`заказ`", db.getConnection());
        table = new DataTable();

        adapted.Fill(table);
        dataGridView1.DataSource = table;
        db.closeConection();
    }

    private void подсчетЗаказавоПоСтатусамToolStripMenuItem_Click(object sender, EventArgs e)
    {
        db = new DataBase();
        db.openConnection();
        adapted = new MySqlDataAdapter("SELECT id_статуса, COUNT(id_статуса) AS
Количество_Статусов FROM `заказ` GROUP BY id_статуса", db.getConnection());
        table = new DataTable();

        adapted.Fill(table);
        dataGridView1.DataSource = table;
        db.closeConection();
    }

    private void отчетыToolStripMenuItem_Click(object sender, EventArgs e)
    {
        //Save_All_report(dataGridView1);
        WebBrowser web = new WebBrowser();
        System.Diagnostics.Process.Start(@"chrome.exe",
"http://127.0.0.1/openserver/phpmyadmin/tbl_export.php?
db=bd_tv_services&table=заказ&single_table=true");
    }
}
}
}

```

## orderForm.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;

namespace WindowsFormsApp1
{
    public partial class statusForm : Form
    {
        private DataBase db = null;
        private DataTable table = null;
        private MySqlDataAdapter adapted = null;

        public statusForm()
        {
            InitializeComponent();
        }

        private void statusForm_Load(object sender, EventArgs e)
        {
            db = new DataBase();
            db.openConnection();
            adapted = new MySqlDataAdapter("SELECT * FROM `статус_заказа`",
db.getConnection());
            table = new DataTable();

            dataGridView1.AllowUserToAddRows = false;
            dataGridView1.AllowUserToDeleteRows = false;
            dataGridView1.ReadOnly = true;

            adapted.Fill(table);
            dataGridView1.DataSource = table;
            db.closeConection();
        }
    }
}

```