

**Міністерство освіти та науки України**  
**Національний університет “Львівська політехніка”**

**Інститут прикладної математики**  
**і фундаментальних наук**  
**Кафедра прикладної математики**

**Курсова робота**  
*з курсу*  
*«Робота з великим базами даних»*

Виконав: студент  
групи ПМ-31  
Вінчура О.А.  
Прийняв:  
Любінський Б.Б.

Львів 2022

## Основні положення

Мета цієї курсової роботи полягає в тому щоб навчитись проектувати великі бази даних та в подальшому працювати з ними через додаткове програмне забезпечення.

Основне завдання полягає в тому щоб розробити базу даних відповідно до предметної області, та автоматизувати роботу з нею засобами програмного забезпечення.

Для створення бази даних використав декларативну мову програмування SQL. SQL (англ. *Structured query language* — мова структурованих запитів) — декларативна мова програмування для взаємодії користувача з базами даних, що застосовується для формування запитів, оновлення і керування реляційними БД, створення схеми бази даних та її модифікації, системи контролю за доступом до бази даних.

За середовище розробки обрав програмний продукт Microsoft SQL Server Manegement Studio.

Для того щоб уможливити та полегшити взаємодію користувача з базою даних розробив застосунок на ієтерфейсі Windows Forms на мові програмування C#, з використанням технології доступу до баз даних Entity Framework.

*Windows Forms* — інтерфейс програмування програм (API), який відповідає за графічний інтерфейс користувача і є частиною Microsoft .NET Framework.

C# — об'єктно-орієнтована мова програмування з безпечною системою типілізації для платформи .NET.

*Entity Framework* — це набір технологій в ADO.NET, які підтримують розробку програмно-орієнтованих на дані програмних додатків. Середовище розробки користувацької програми – Microsoft Visual Studio.

Для створення додатку обрав принцип *Database First*, який полягає в тому що ми створюємо структуру бази даних самі, а потім імпортуємо наявну базу даних в модель, на основі якої Entity Framework генеруватиме необхідні класи та подбає про відображення.

## **Предметна область**

### ***Варіант 4. "Екскурсії"***

*Інформаційна система служить для організації, яка проводить автобусні екскурсії на замовлення.*

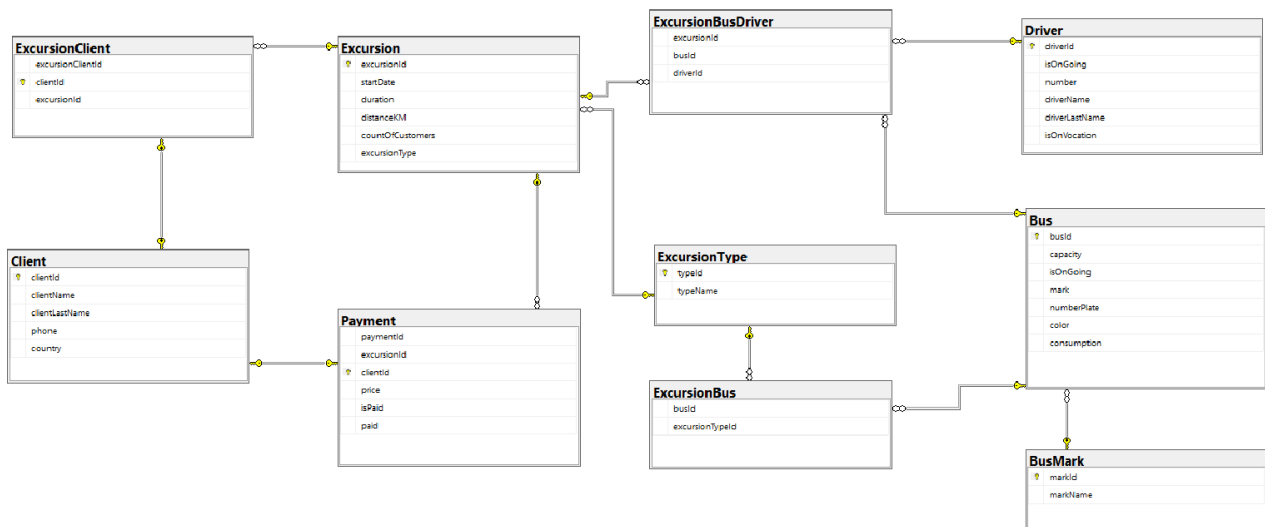
Кожна екскурсія є запланованою на певний день, має певну тривалість, місце призначення, відстань, замовника, тип, кількість екскурсантів. На кожну екскурсію плануються певні автобуси які мають певну місткість, шофера, можуть обслуговувати певні типи екскурсій, характеризуються розходом бензину на кілометр. За кожну екскурсію система повинна нарахувати оплату за певною формулою яка враховує кількість екскурсантів, розхід бензину та тривалість екскурсії. Система повинна враховувати надходження платежів за екскурсії згідно виставлених рахунків. Автобус та шофер не можуть одночасно обслуговувати дві екскурсії. Екскурсія не може надаватись замовнику, який має борг на протязі 30 днів.

**Система повинна надавати наступні звіти:**

- Список екскурсій за певний проміжок часу з вказанням вартості.
- Графік використання автобусів за певний проміжок часу.
- Фінансовий звіт за місяць, який вказує для кожного дня загальну суму виконаних екскурсій та надходження оплати.
- Довідку замовника з вказанням замовлених екскурсій та проведених оплат.

# Структура бази даних

Створив базу даних відповідно до предметної області:



Згодом наповнив її відповідними даними на свій розсуд.

Створив відповідні процедури та тригери для контролю бази даних.

--За кожну екскурсію система повинна нарахувати оплату за певною формулою  
--яка враховує кількість екскурсантів, розхід бензину та тривалість екскурсії

```
CREATE OR ALTER FUNCTION sum_price(@exc_id INT) RETURNS MONEY
AS
BEGIN
    DECLARE @cust_count INT, @gas_consumption FLOAT, @exc_duration FLOAT, @dist FLOAT,
    @excPrice MONEY;
    SELECT @cust_count = Excursion.countOfCustomers, @gas_consumption = Bus.consumption,
    @exc_duration = Excursion.duration, @dist = Excursion.distanceKM, @excPrice =
    (@dist/@gas_consumption)*51
    FROM Excursion JOIN ExcursionBusDriver ON
    Excursion.excursionId=ExcursionBusDriver.excursionId JOIN Bus ON ExcursionBusDriver.busId =
    Bus.busId
    WHERE Excursion.excursionId = @exc_id
    IF @exc_duration = 0 RETURN 0;
    IF @exc_duration > 0 RETURN (@excPrice + @cust_count*@excPrice*20/100 +
    @exc_duration*300)
    RETURN 0
END
GO

GO
SELECT Excursion.excursionId, dbo.sum_price(Excursion.excursionId) AS sum
FROM Excursion join ExcursionBusDriver on
Excursion.excursionId=ExcursionBusDriver.excursionId join Bus on ExcursionBusDriver.busId =
Bus.busId
GO
```

--процедура для заповнення чеків за певний час, яка викликає функцію sum\_price

```
DROP PROCEDURE IF EXISTS price_for_excursions
CREATE PROCEDURE pay_for_excursions (@startDate DATE)
AS DECLARE paymentCursor CURSOR LOCAL FOR
```

```

SELECT paymentId FROM Payment
JOIN Excursion ON Excursion.excursionId = payment.excursionId
WHERE startDate BETWEEN @startDate AND DATEADD(month, 1, @startDate)
DECLARE @paymentId INT
OPEN paymentCursor
FETCH NEXT FROM paymentCursor INTO @paymentId
WHILE @@FETCH_STATUS=0
BEGIN

    UPDATE Payment
    SET price = dbo.sum_price(payment.excursionId)
    WHERE paymentId = @paymentId

FETCH NEXT FROM paymentCursor INTO @paymentId
END
CLOSE paymentCursor

```

---

--процедура оплати чеків

```

DROP PROCEDURE IF EXISTS pay_for_excursions
CREATE PROCEDURE pay_for_excursions (@startDate DATE)
AS DECLARE paymentCursor CURSOR LOCAL FOR
SELECT paymentId FROM Payment
JOIN Excursion ON Excursion.excursionId = payment.excursionId
WHERE startDate BETWEEN @startDate AND DATEADD(month, 1, @startDate)
DECLARE @paymentId INT
OPEN paymentCursor
FETCH NEXT FROM paymentCursor INTO @paymentId
WHILE @@FETCH_STATUS=0
BEGIN

    UPDATE Payment
    SET paid = price, isPaid = 1
    WHERE paymentId = @paymentId

FETCH NEXT FROM paymentCursor INTO @paymentId
END
CLOSE paymentCursor

```

---

-- Екскурсія не може надаватись замовнику, який має борг впродовж 30 днів.  
 --(тригер для таблиці Payment, так як немає чеку немає і екскурсії)

```

CREATE TRIGGER debt_limit
ON Payment
INSTEAD OF INSERT
AS
BEGIN
    DECLARE @haveDebt INT
    DECLARE @clientId INT

    SELECT @clientId = clientId FROM inserted

    SET @haveDebt = (SELECT count(*) FROM Payment JOIN Excursion ON Excursion.excursionId
    = Payment.excursionId
    WHERE clientId = @clientId AND startDate < DATEADD(month, -1, CURRENT_TIMESTAMP))

    IF @haveDebt <= 0
        INSERT INTO payment
        SELECT * FROM inserted
    ELSE
        SELECT 'Client has a debt' AS 'DEBT'

END

```

---

```

--тригер для таблиці ExcursionBusDriver
--Автобус та шофер не можуть одночасно обслуговувати дві екскурсії

CREATE TRIGGER driverBusLimit
ON ExcursionBusDriver
INSTEAD OF INSERT
AS
BEGIN
    DECLARE @onGoingBus INT
    DECLARE @onGoingDriver INT
    DECLARE @driverId INT
    DECLARE @busId INT

    SELECT @driverId = driverId FROM inserted
    SELECT @busId = busId FROM inserted

    SET @onGoingDriver = (SELECT count(*) FROM ExcursionBusDriver
                        JOIN Excursion ON Excursion.excursionId=
ExcursionBusDriver.excursionId
                        WHERE driverId = @driverId AND CURRENT_TIMESTAMP BETWEEN
startDate AND DATEADD(day, duration, startDate))

    SET @onGoingBus = (SELECT count(*) FROM ExcursionBusDriver
                        JOIN Excursion ON Excursion.excursionId=
ExcursionBusDriver.excursionId
                        WHERE busId = @busId AND CURRENT_TIMESTAMP BETWEEN
startDate AND DATEADD(day, duration, startDate))

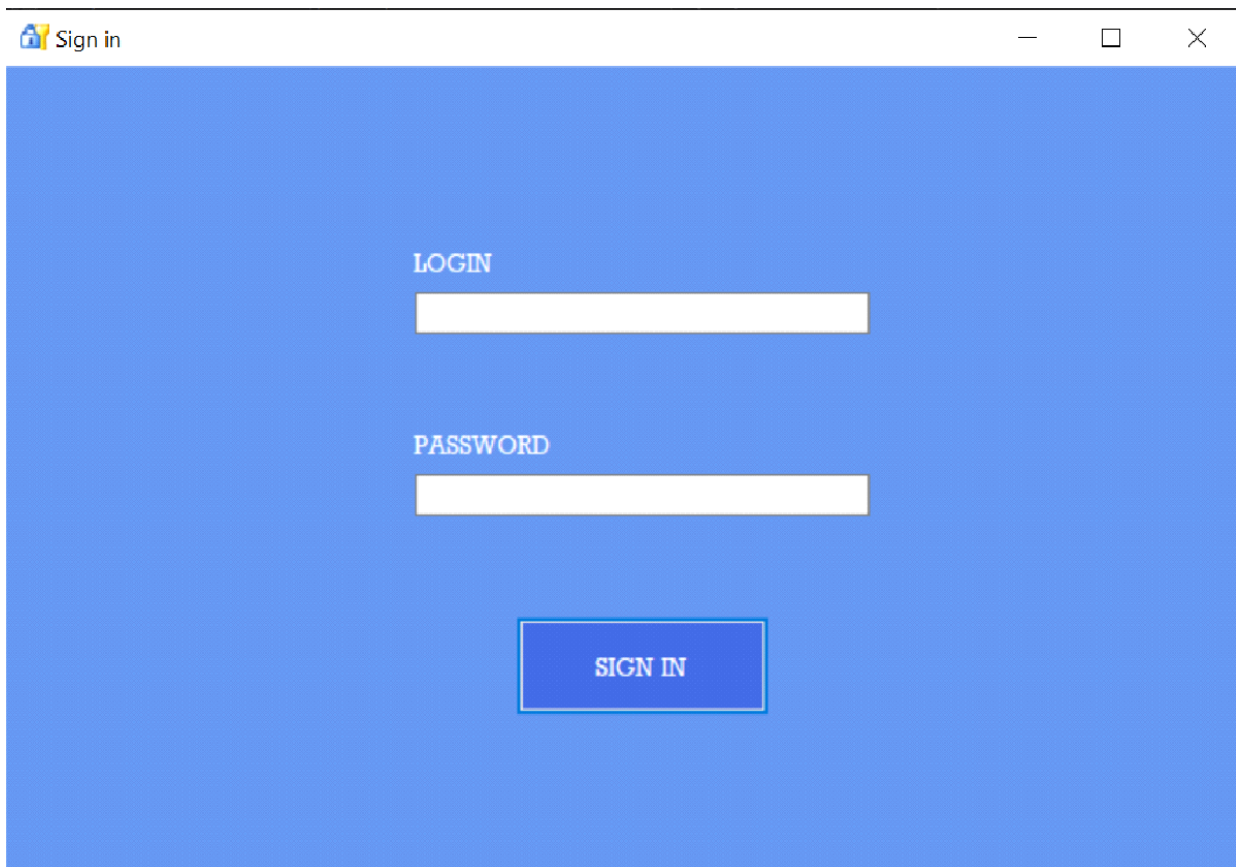
    IF @onGoingBus > 0
        SELECT 'Bus is Busy' AS 'BUS'
    ELSE IF @onGoingDriver > 0
        SELECT 'Driver is Busy' As 'Driver'
    ELSE
        INSERT INTO ExcursionBusDriver
        SELECT driverId, busId FROM inserted
END
GO

```

## Огляд застосунку

Створив клієнтський додаток з графічним інтерфейсом на Windows Forms на основі Entity Framework. Для зручності розробки також використовував систему контролю версій Git.

Після запуску програми, користувача зустрічає вікно авторизації, де потрібно ввести логін та пароль користувача, щоб отримати доступ перегляду до бази даних. При введенні невірних даних входу програма не дає доступу до функціоналу.



The screenshot shows a Windows Forms application window titled "Sign in". The window has a standard Windows title bar with minimize, maximize, and close buttons. The main content area has a solid blue background. In the center, there is a login form consisting of two white text boxes with black borders. The first text box is labeled "LOGIN" and the second is labeled "PASSWORD". Below these text boxes is a blue button with white text that says "SIGN IN".

Після вдалої авторизації користувач потрапляє у головне меню програми. Тут містяться вкладки з усіма таблицями, які при необхідності можна відредагувати включно з функцією додавання та видалення елементів з таблиць.

Excursion db

Excursions	Clients	Payments	Buses	Drivers	ExcursionBusDriver	ExcursionClient	ExcursionBus	Income statistics
------------	---------	----------	-------	---------	--------------------	-----------------	--------------	-------------------

1 of 7

excursionId	startDate	duration	distanceKM	countOfCustomers	excursionType
1	01.03.2022	4,5	300	53	3
2	05.03.2022	8,5	520	70	4
3	09.03.2022	1,5	45	9	2
4	02.04.2022	6,5	340	70	4
5	10.05.2022	2	70	9	1
6	10.05.2022	2,5	20	75	2
7	10.06.2022	3,5	210	50	3

До кожної таблиці також створений пошук за відповідними фільтрами:

Excursion db

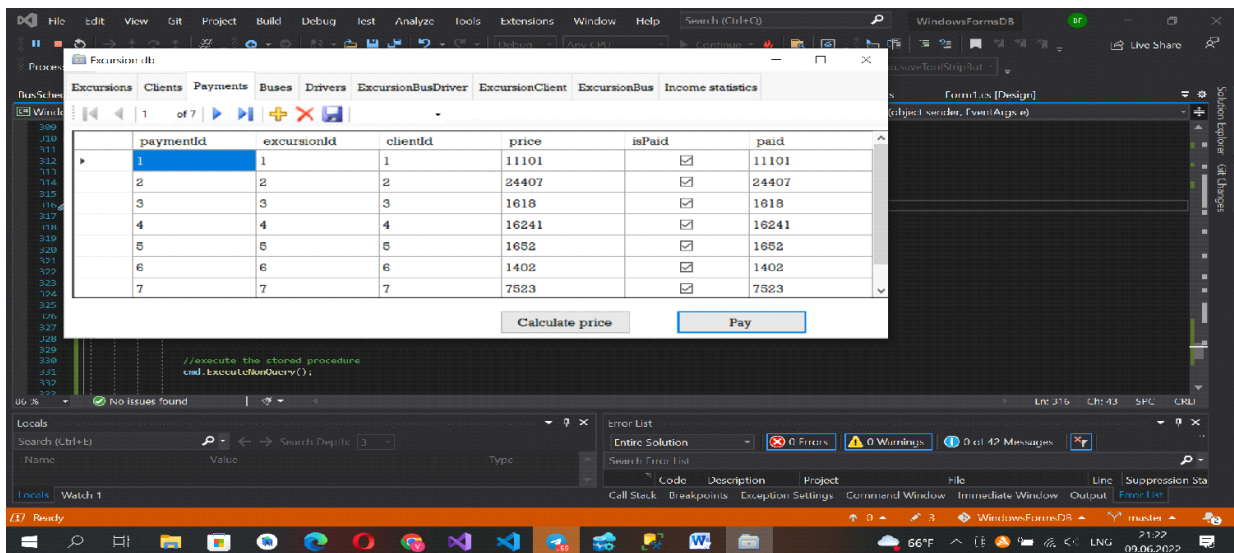
Excursions	Clients	Payments	Buses	Drivers	ExcursionBusDriver	ExcursionClient	ExcursionBus	Income statistics
------------	---------	----------	-------	---------	--------------------	-----------------	--------------	-------------------

1 of 7

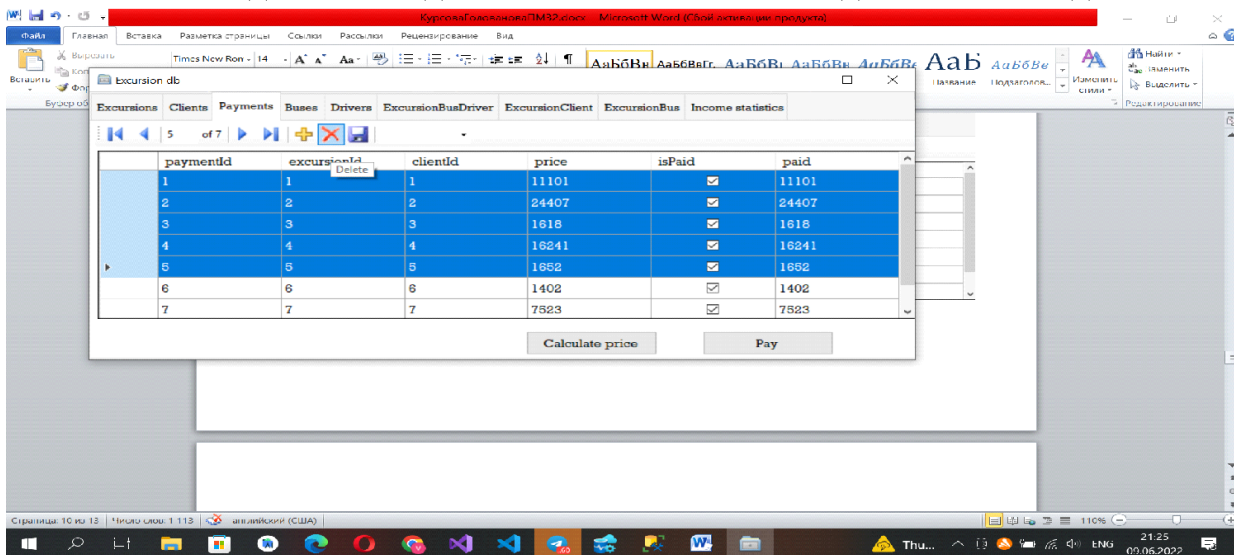
clientId	clientName	driverName	phone	country
1	Anna	Pylyp	0969999991	38
2	Oleksiy	Shpak	0969999992	38
3	Jameson	Wels	0969999993	38
4	Kamil	Nowak	0969999994	38
5	Oleksandr	Koval	0969999995	38
6	Maria	Koval	0969999996	38
7	Volodymyr	Ivashko	0969999997	38

Автоматизація роботи присутня у вигляді кнопок для деяких таблиць:





Також є видалення декількох елементів одночасно виділяючи їх:



Введення даних контролюється та валідується за потреби

Excursion db

Excursions	Clients	Payments	Buses	Drivers	ExcursionBusDriver	ExcursionClient	ExcursionBus	Income statistics
------------	---------	----------	-------	---------	--------------------	-----------------	--------------	-------------------

1 of 12

	busId	capacity	isOnGoing	mark	numberPlate	color	consumption
▶	1	7	<input type="checkbox"/>	Renault	BC1111BE	white	7,1
	2	71	<input type="checkbox"/>	Renault	BC1112AC	black	18,2
	3	9	<input type="checkbox"/>	Citroen	BC1113AE	red	5,5
	4	9	<input type="checkbox"/>	Ford	BC1114CE	white	9,5
	5	55	<input type="checkbox"/>	Mercedes-Benz	BC1115AA	gray	10,8
	6	15	<input type="checkbox"/>	Scania	BC1116BA	black	30,6
	7	9	<input type="checkbox"/>	Fiat	AA1116BE	white	9,9

Після редагування таблиць, можемо зберегти результат, за допомогою кнопки “Save” тоді дані перенесуться до бази даних:

Excursion db

Excursions	Clients	Payments	Buses	Drivers	ExcursionBusDriver	ExcursionClient	ExcursionBus	Income statistics
------------	---------	----------	-------	---------	--------------------	-----------------	--------------	-------------------

4 of 13

	driverId	isOnGoing	Save	number	driverName	driverLastName	isOnVocation
	1	<input type="checkbox"/>		+380666666661	Vasyl	Vasylenko	<input type="checkbox"/>
	2	<input type="checkbox"/>		+380666666662	Ivan	Ivanenko	<input type="checkbox"/>
	3	<input type="checkbox"/>		+380666666663	Taras	Tarassenko	<input type="checkbox"/>
▶	4	<input type="checkbox"/>		+380666666664	Stepan	Stepanenko	<input type="checkbox"/>
	5	<input type="checkbox"/>		+380666666665	Andriy	Shevchenko	<input type="checkbox"/>
	6	<input type="checkbox"/>		+380666666666	Kyrylo	Melnyk	<input type="checkbox"/>
	7	<input type="checkbox"/>		+380666666667	Mykyta	Boiko	<input type="checkbox"/>

Окрему роль відіграє остання вкладка «Income statistics»,

Excursion db

Excursions	Clients	Payments	Buses	Drivers	ExcursionBusDriver	ExcursionClient	ExcursionBus	Income statistics
------------	---------	----------	-------	---------	--------------------	-----------------	--------------	-------------------

Excursions for a certain period

Bus usage schedule

Customer's payment certificates

Financial report for the month

де містяться необхідні звіти, наприклад:

- Список екскурсій за певний проміжок часу з вказанням вартості:

ExcursionForPeriod

List of excursions for a certain period of time with price

Start date  
1 квітня 2022 р.

End date  
6 червня 2022 р.

Search

	excursionId	startDate	duration	distanceKM	countOfCustomers	excursionType	Payment
▶	4	02.04.2022	6.5	340	70	4	3700
	5	10.05.2022	2	70	9	1	7500
	6	10.05.2022	2.5	20	75	2	900

- Графік використання автобусів за певний проміжок часу:

BusSchedule

Bus usage schedule for a certain period of time

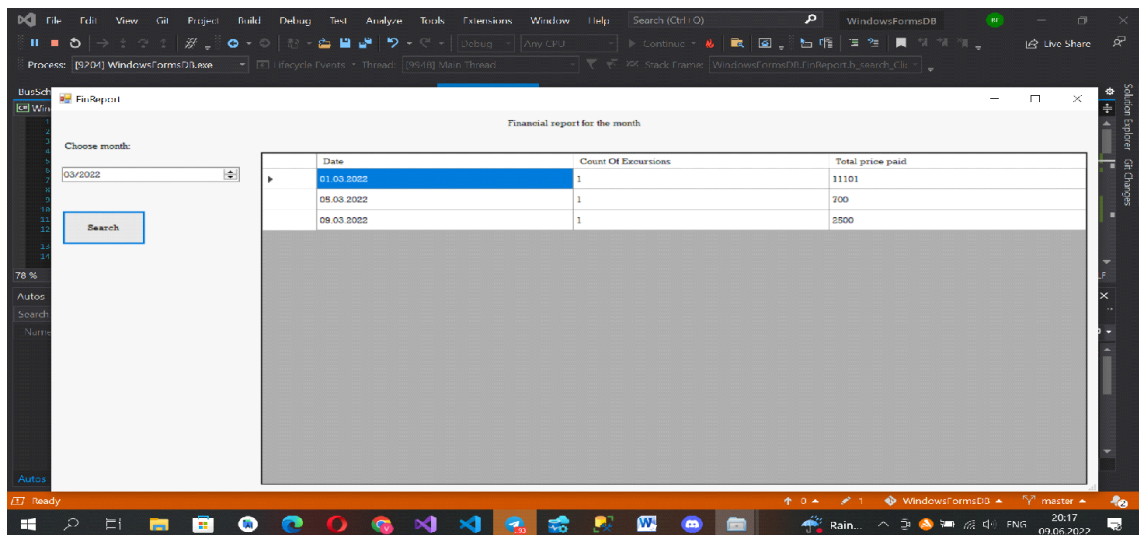
Start date  
1 березня 2022 р.

End date  
1 травня 2022 р.

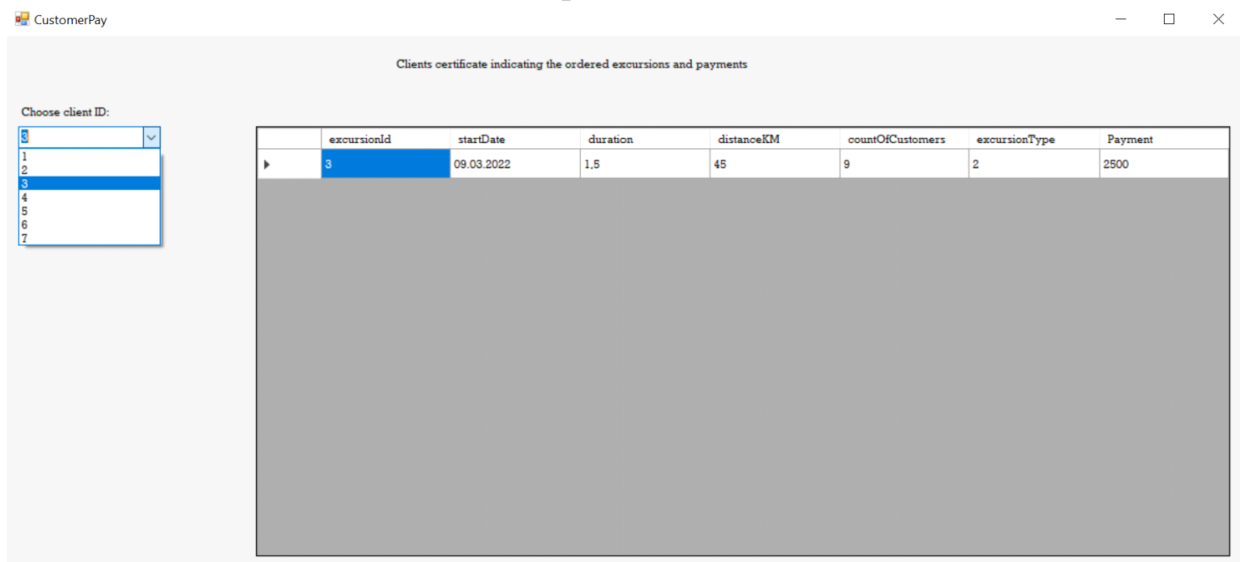
Search

	Date	excursionId	busId	driverId
▶	05.03.2022	2	2	2
	09.03.2022	3	3	3
	02.04.2022	4	2	4

- Фінансовий звіт за місяць, який вказує для кожного дня загальну суму надходження оплати та виконаних екскурсій:



- Довідку по номеру замовника з вказанням замовлених ним екскурсій та відповідно проведених оплат:



Для кожного інформаційного звіту відкривається окрема форма (вікно).

Для зручності використання додатку всі поля вводу проходять валідацію.

## Список використаних джерел

- Стаття про Database First [Електронний ресурс] – Режим доступу:  
<https://docs.microsoft.com/en-us/ef/ef6/modeling/designer/workflows/database-first>
- Стаття про SQL [Електронний ресурс] – Режим доступу:  
[https://uk.wikipedia.org/wiki/SQL#%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0\\_SQL](https://uk.wikipedia.org/wiki/SQL#%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0_SQL)
- Стаття про C Sharp [Електронний ресурс] – Режим доступу:  
[https://uk.wikipedia.org/wiki/C\\_Sharp](https://uk.wikipedia.org/wiki/C_Sharp)
- Стаття про EF Core [Електронний ресурс] – Режим доступу:  
<https://docs.microsoft.com/en-us/ef/core/get-started/overview/first-app?tabs=netcore-cli>