Міністерство освіти і науки України

Одеський національний політехнічний університет

Інститут комп’ютерних систем

Кафедра ком’ютеризованих систем управління і автоматики

Розрахунково-графічна робота

з дисципліни: «Програмне забезпечення сучасних ЕОМ»

Виконав:

ст. гр. АТ-132  
Кравченко І. А.

Перевірив:

Положаєнко С. А.

Одеса 2016

1. Вступ

Метою даної РГР є поглиблення знань в області нових комп'ютерних інформаційних технологій, отримання досвіду самостійно створювати структуру бази даних, встановлювати зв'язки між таблицями, створювати форми для введення даних, запити для отримання вибірок даних і форми для їх виведення, формувати звіти.

На даний час реляційні бази даних широко використовуються для збереження та обробки великих масивів даних. Крім того, має бути забезпечений оперативний доступ до цих даних.

Реляційна база даних — [база даних](https://uk.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85), заснована на [реляційній моделі даних](https://uk.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BB%D1%8F%D1%86%D1%96%D0%B9%D0%BD%D0%B0_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85" \o "Реляційна модель даних). Слово «реляційний» походить від [англ.](https://uk.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D1%96%D0%B9%D1%81%D1%8C%D0%BA%D0%B0_%D0%BC%D0%BE%D0%B2%D0%B0) *relation*. Для роботи з реляційними БД застосовують [реляційні СКБД](https://uk.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BB%D1%8F%D1%86%D1%96%D0%B9%D0%BD%D0%B0_%D0%A1%D0%9A%D0%91%D0%94" \o "Реляційна СКБД). Інакше кажучи, реляційна база даних — це база даних, яка сприймається користувачем як набір нормалізованих відношень різного ступеня. У реляційній моделі досягається більш високий рівень абстракції даних, ніж в ієрархічній або мережевій.

Вибір прикладної задачі для вирішення в контрольній роботі проводиться студентом самостійно і узгоджується з викладачем.

Прикладною задачею було обрано створення системи управління базою даних для фотографа. Це має бути програма для операційної системи Windows, написана мовою програмування C# на платформі .Net Framework. База даних має містити інформацію про клієнтів та замовлення фотозйомок. Вона створена в СКБД Miscrosoft SQL Server. Доступ до неї з програми виконується за допомогою ORM (Object Relational Mapping) Entity Framework. Це об’єктно-орієнтована технологія доступу до даних, що транслює інформацію з бази даних в об’єкти в програмі.

Програма створювалась з використанням форм WinForms Controls компанії DevExpress. Ця компанія розробляє рішення для розробників прикладного програмного забезпечення, а саме форми для користувальницького інтерфейсу як на десктопні застосунки, так і для веб-сайтів.

1. Проект бази даних

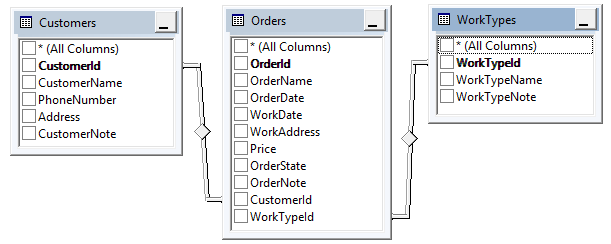
База даних має три таблиці: «Customers» - таблиця клієнтів, «Orders» - таблиця замовлень, «WorkTypes» - таблиця типів робіт.

Рисунок 1 – Таблиці «Customers», «Orders», «WorkTypes»

* 1. Таблиця «Customers»

Таблиця «Customers» має такі поля:

* ключове поле CustomerId;
* текстове поле CustomerName, яке містить ім’я клієнта;
* текстове поле PhoneNumber, що містить номер телефону;
* текстове поле Address, де зберігається адреса клієнта;
* текстове поле CustomerNote, що містить примітку або пояснення про клієнта.
  1. Таблиця «Orders»

Ця таблиця містить інформацію про замовлення робіт та має такі поля:

* ключове поле OrderId;
* текстове поле OrderName, яке зберігає назву замовлення;
* поля типу дата OrderDate, що зберігає дату замовлення, та WorkDate, що зберігає дату проведення зйомки;
* текстове поле WorkAddress, яке зберігає адресу проведення зйомки;
* числове поле Price, що містить ціну замовлення;
* поле OrderState, яке означає стан замовлення: 0 – виконується, 1 – виконаний, 2 – скасований;
* тексове поле OrderNote – примітка про замовлення;
* числові поля CustomerId, що містить посилання на клієнта та WorkTypeId, що містить посилання на тип роботи;
  1. Таблиця «WorkTypes»

Ця таблиця має містити інформацію про типи замовлень. Наприклад, фото на паспорт, відеозйомка весілля, портрет, фотозйомка віньєток та інші. Містить такі поля:

* ключове поле WorkTypeId;
* текстове поле WorkTypeName, яке має назву роботи (фото на паспорт, портрет та інше);
* текстове поле WorkTypeNote - примітка про тип роботи.

1. Вибір засобів програмування застосунка

Для створення додатка була обрана мова програмування C# та платформа .Net Framework. Програма представляє собою клієнтський застосунок зі зручним інтерфейсом на ОС Windows, що надає доступ до бази даних та має вже сформовані запити.

Для створення бази даних був обраний Microsoft SQL Server як потужний засіб зберігання даних.

Доступ до неї з програми виконується за допомогою технології ORM (Object Relational Mapping) Entity Framework. Це об’єктно-орієнтована технологія доступу до даних, що транслює інформацію з бази даних в об’єкти в програмі. Використовувався принцип Code First. Це означає, що спочатку були сформовані класи-сутності (Customer, Order, WorkType), потім створений клас PmContext, що має в собі списки цих сутностей і надає доступ до бази даних. Класи в програмі створюються в базі даних як таблиці, а властивості класів – як поля таблиць.

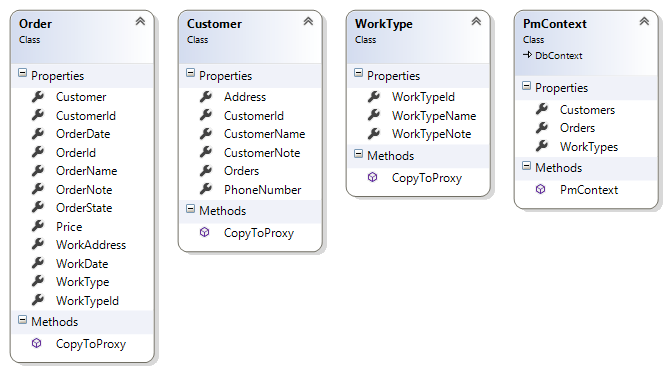


Рисунок 2 - Діаграма класів Order, Customer, WorkType, PmContext

Запити до бази даних можна робити за допомогою двох механізмів:

* SQL-запити;
* LINQ To SQL-запити.

SQL-запит представляє собою звичайний SQL-запит всередині програмного коду. Наприклад:

var customers = pmContext.Customers.SqlQuery("select \* from dbo.Customers").ToList();

pmContext – посилання на об’єкт класу PmContext, що надає доступ до бази даних; він має властивість Customers, що представляє собою колекцію сутностей Customers. SqlQuery – метод, що виконує sql-запит, та повертає колекцію об’єктів з бази даних. Даний механізм працює швидше, ніж LINQ To SQL-запит, але має один великий недолік: такий програмний код не можна назвати безпечним. Тому що сам SQL-запит передається у метод як символьний рядок і не може бути перевірений на помилки до компіляції програми. Якщо програміст зробить помилку у якомусь слові, виникне Runtime Exception, що означає помилку часу виконання.

По-іншому працюють LINQ To SQL-запити. Ці запити відповідають усім нормам об’єктно-орієнтованого програмування, а також повністю перевіряються компілятором ще до компіляції програми. Кожен LINQ To SQL-запит під час виконання транслюється у SQL-запит, а потім виконується до бази даних. Це дозволяє використовувати об’єкти у таких запитах, тому не може бути непомічених помилок. Хоча такі запити виконуються трохи довше ніж SQL-запити, але вони є зручним засобом обміну даними із базою даних.

Приклад LINQ To SQL-запита.

IQueryable<PrxCustomer> getList = from qr in pmContext.Customers

select new PrxCustomer

{

Id = qr.CustomerId,

Name = qr.CustomerName,

Address = qr.Address,

Note = qr.CustomerNote,

PhoneNumber = qr.PhoneNumber

};

List<PrxCustomer> customers = getList.ToList();

Даний запит є аналогом попереднього SQL-запиту, але він дозволяє працювати з технологією IntelliSense, що значно зменшує кількість помилок.

1. Використання додатка

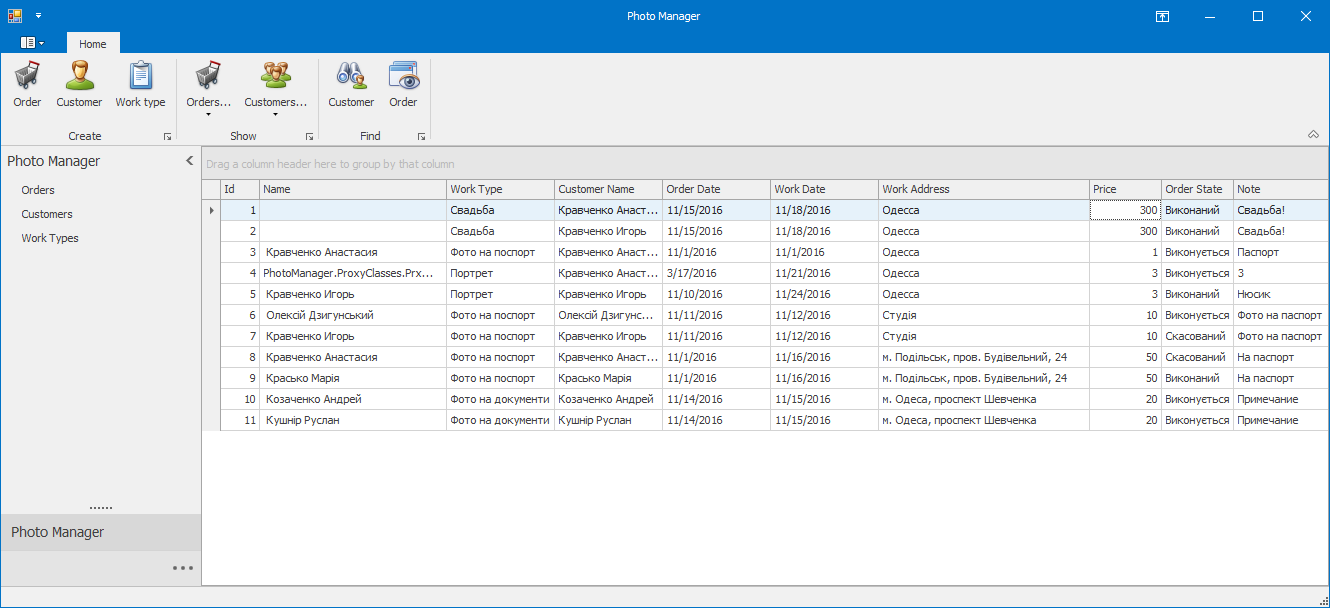
Коли користувач запускає програму, виникає головна форма з таблицею замовлень по центру. Зліва знаходиться дерево навігації. Усі дані відображаються у таблицях по центру. Зверху знаходиться панель управління. Щоб проглядати списки клієнтів, замовлень та типів робот, можна користуватись деревом навігації.

Рисунок 3 – Головне вікно програми зі списком замовлень

Можна групувати дані. Наприклад, можна згрупувати замовлення за типом роботи:

Рисунок 4 – Групування за типом роботи

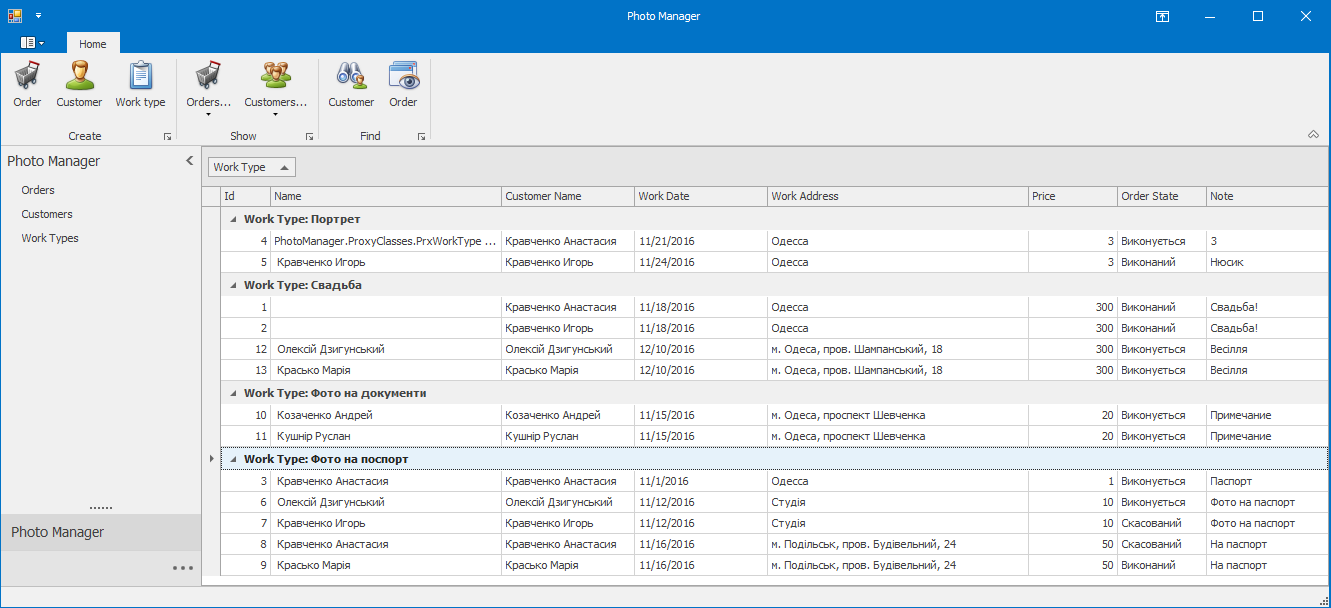


Рисунок 5 – Групування за типом роботи

Можна використовувати складне групування з багатьма параметрами. Наприклад, групування за типом роботи та станом замовлення:

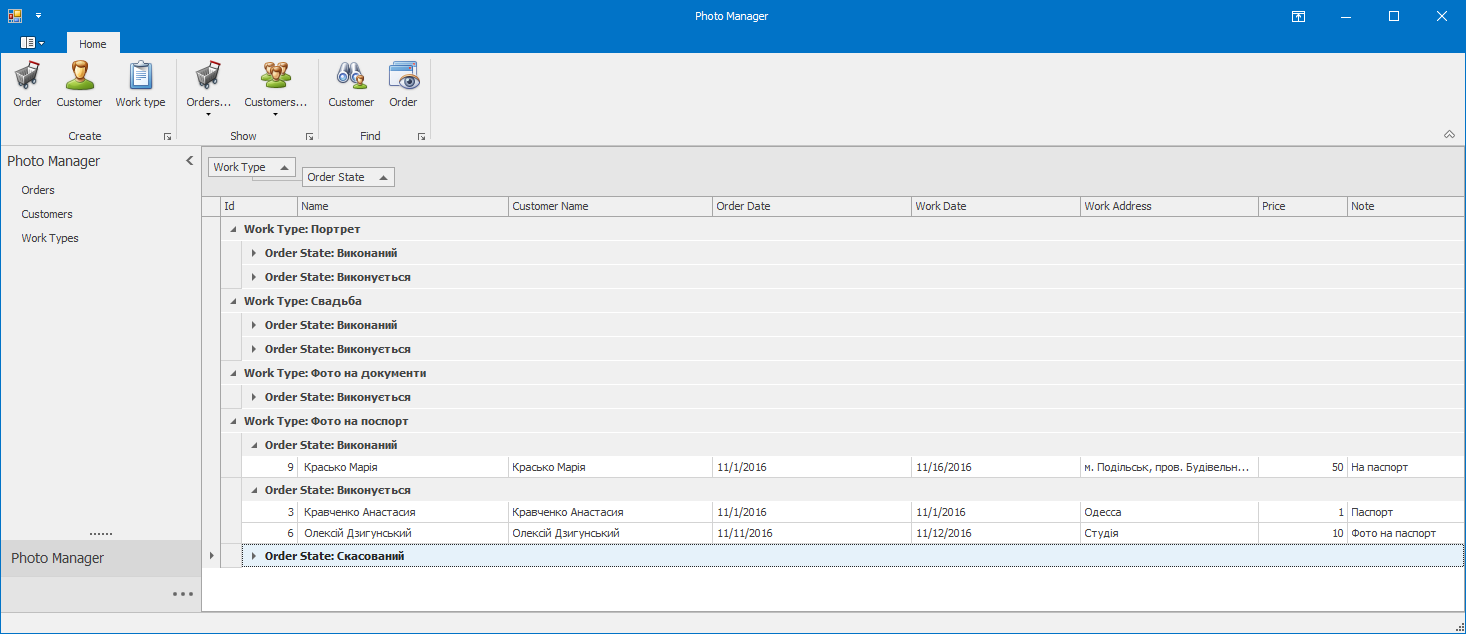


Рисунок 6 - Групування за двома параметрами

Щоб побачити список клієнтів, потрібно натиснути в дереві навігації кнопку «Customers»:

Рисунок 7 – Список клієнтів

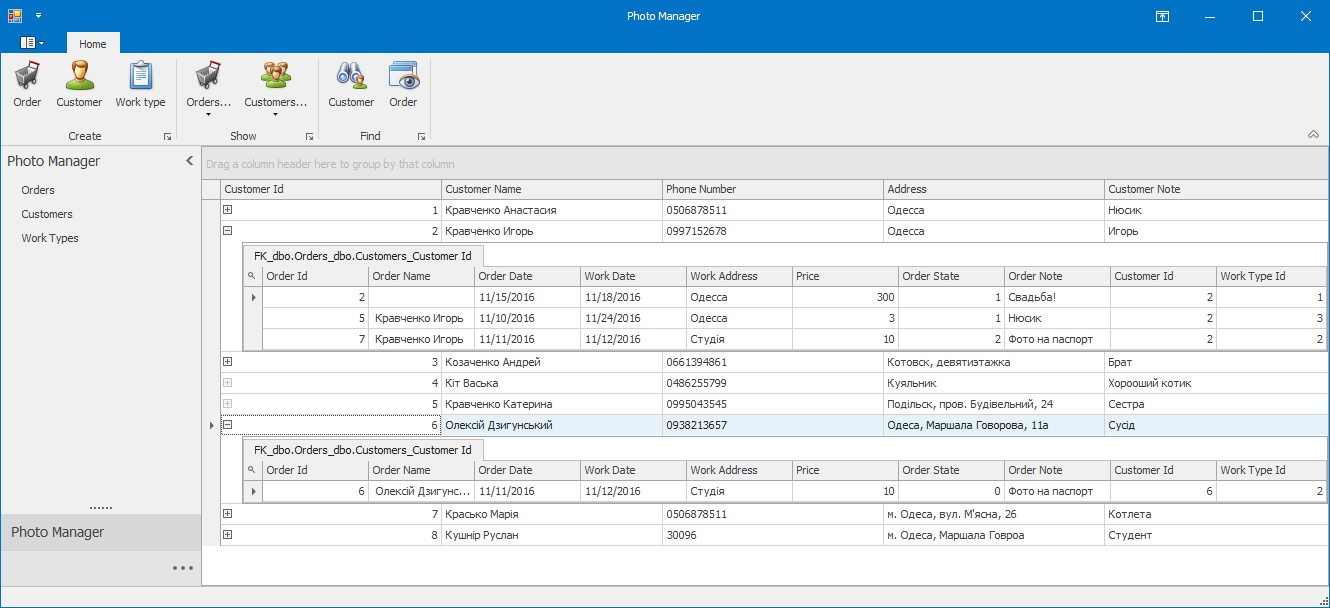


Рисунок 8 – Список клієнтів

Аналогічно для типів роботи. Потрібно натиснути кнопку «Work types», щоб побачити список типів роботи.

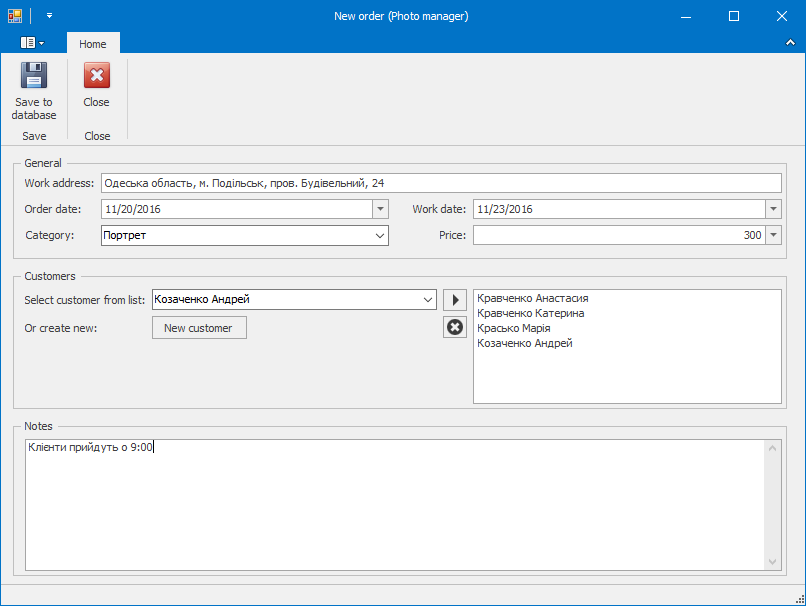
Щоб додати нове замовлення, потрібно натиснути кнопку «Order» у групі «Create». Виникне нове вікно створення замовлення:

Рисунок 9 – Вікно створення замовлення

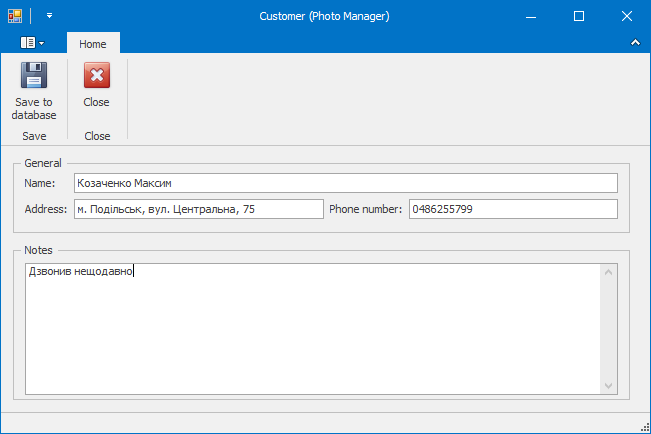
Можна обрати декількох клієнтів з випадаючого списку. Якщо це новий клієнт його модна додати, натиснувши кнопку «New customer», або натиснувши в головній формі кнопку «Customer» у групі «Create». Виникне вікно створення клієнта:

Рисунок 10 – Вікно додавання клієнта

Користувач може додавати типи роботи. Під час написання програми була думка створити декілька основних типів роботи (віньєтка, фотосесія, фото на паспорт, портрет, зйомка весілля), але таких типів роботи може бути дуже багато, тому що фотограф зазвичай робить багато різної роботи. Тому було вирішено створити окрему таблицю в базі даних і дати змогу користувачу самому вирішувати, які типи робіт йому потрібні і додавати їх, якщо

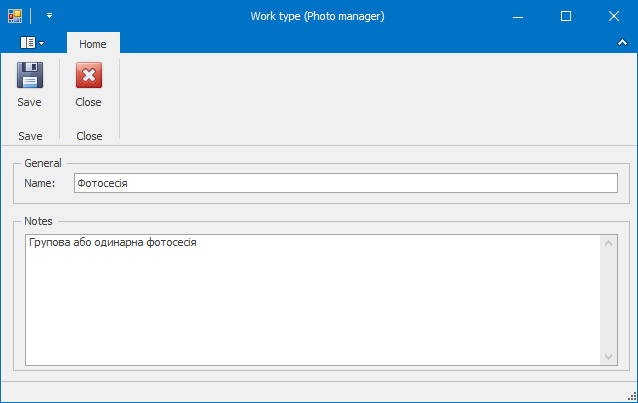
Для додавання нового типу роботи, потрібно натиснути кнопку «Work type» у групі «Create». Виникне відповідне вікно:

Рисунок 11 – Вікно створення типу роботи

Висновки

Метою даної розрахунково-графічної роботи було поглиблення знань в області нових комп'ютерних інформаційних технологій, отримання досвіду самостійно створювати структуру бази даних, встановлювати зв'язки між таблицями, створювати форми для введення даних, запити для отримання вибірок даних і форми для їх виведення, формувати звіти.

В ході розрахунково-графічної роботи була написана програма мовою програмування C#, що надає доступ до бази даних, створеної за допомоги ORM Entity Framework в SQL Server. Користувач програми має змогу додавати нові замовлення, клієнтів, типи робот, викладачів. Програма має готові запити до бази даних. Застосунок був написаний згідно із завданням. Усі потреби були задовільнені.

Список використаної літератури

1. Використання Code First для створення бази даних - https://msdn.microsoft.com/ru-ru/data/jj193542(v=vs.113).aspx
2. Документація з використання Data Grid Control https://documentation.devexpress.com/#WindowsForms/CustomDocument3455
3. [К. Дж. Дейт](https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D0%B9%D1%82,_%D0%9A%D1%80%D0%B8%D1%81%D1%82%D0%BE%D1%84%D0%B5%D1%80). Введение в системы баз данных = Introduction to Database Systems. — 8-е изд. — М.: Вильямс, 2006. — С. 1328. — [ISBN 5-8459-0788-8](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D1%83%D0%B6%D0%B5%D0%B1%D0%BD%D0%B0%D1%8F:%D0%98%D1%81%D1%82%D0%BE%D1%87%D0%BD%D0%B8%D0%BA%D0%B8_%D0%BA%D0%BD%D0%B8%D0%B3/5845907888).
4. Керівництво з Entity Framework - <http://metanit.com/sharp/entityframework/>
5. Джеффри Рихтер, Мартен ван де Боспурт. WinRT: программирование на C# для профессионалов = Windows Runtime via C#. — М.: [Вильямс](https://ru.wikipedia.org/w/index.php?title=%D0%92%D0%B8%D0%BB%D1%8C%D1%8F%D0%BC%D1%81_(%D0%B8%D0%B7%D0%B4%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D1%81%D1%82%D0%B2%D0%BE)&action=edit&redlink=1), 2014. — 368 с. — [ISBN 978-5-8459-1910-6](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D1%83%D0%B6%D0%B5%D0%B1%D0%BD%D0%B0%D1%8F:%D0%98%D1%81%D1%82%D0%BE%D1%87%D0%BD%D0%B8%D0%BA%D0%B8_%D0%BA%D0%BD%D0%B8%D0%B3/9785845919106).
6. Джеффри Рихтер. CLR via C#. Программирование на платформе Microsoft .NET Framework 4.5 на языке C#. — 4-е изд. — СПб.: [Питер](https://ru.wikipedia.org/wiki/%D0%9F%D0%B8%D1%82%D0%B5%D1%80_(%D0%B8%D0%B7%D0%B4%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D1%81%D1%82%D0%B2%D0%BE)), 2013. — 896 с. — [ISBN 978-5-496-00433-6](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D1%83%D0%B6%D0%B5%D0%B1%D0%BD%D0%B0%D1%8F:%D0%98%D1%81%D1%82%D0%BE%D1%87%D0%BD%D0%B8%D0%BA%D0%B8_%D0%BA%D0%BD%D0%B8%D0%B3/9785496004336).
7. C# 4.0: полное руководство = C# 4.0 The Complete Reference. — М.: «Вильямс», 2011. — 1056 с. — [ISBN 978-5-8459-1684-6](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D1%83%D0%B6%D0%B5%D0%B1%D0%BD%D0%B0%D1%8F:%D0%98%D1%81%D1%82%D0%BE%D1%87%D0%BD%D0%B8%D0%BA%D0%B8_%D0%BA%D0%BD%D0%B8%D0%B3/9785845916846).
8. Боуман Д., Эмерсон С., Дарновски М. Практическое руководство по SQL. / 4–е изд.: Пер. с англ. – М.: Изд. дом “Вильямс”, 2001. – 352 с.