**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**ДВНЗ «УНІВЕРСИТЕТ БАНКІВСЬКОЇ СПРАВИ»**

**ІНСТИТУТ БАНКІВСЬКИХ ТЕХНОЛОГІЙ ТА БІЗНЕСУ**

**КАФЕДРА КІБЕРБЕЗПЕКИ ТА СОЦІАЛЬНИХ НАУК**

**КУРСОВА РОБОТА**

з дисципліни

**«Програмування (Рівень В- Об’єктно-орієнтовне програмування)»**

на тему: **Дизайн квартир**

Студента(ки) курсу групи

спеціальності «Кібербезпека»

. Бурий Ю.А. .

(прізвище та ініціали)

Керівник

к.т.н., доцент .

(посада, вчене звання, науковий ступінь,

Гордєєв О.О. .

прізвище та ініціали)

Національна шкала \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів: \_\_\_\_

Оцінка: ECTS \_\_\_\_

Члени комісії

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

Київ - 2019 рік

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**ДВНЗ «УНІВЕРСИТЕТ БАНКІВСЬКОЇ СПРАВИ»**

**ІНСТИТУТ БАНКІВСКЬИХ ТЕХНОЛОГІЙ ТА БІЗНЕСУ**

**КАФЕДРА КІБЕРБЕЗПЕКИ ТА СОЦІАЛЬНИХ НАУК**

Спеціальність: «Кібербезпека»

Курс Група 107-Кбс

Завдання

На курсову роботу студента

Бурого Юрія

1. Тема курсової роботи: Дизайн квартир
2. Термін здачі курсової роботи студентом: 12.12.2019
3. Вихiднi данi до (роботи): інформаційно-аналітичні інтернет джерела, зразки рішень практичних завдань та задач, нормативно-правова база, щодо оформлення технічної документації програмних продуктів, початкові значення для обробки.
4. Змiст пояснювальної записки (перелiк питань, які належить розробити) Вступ. Формулювання вимог до програми. Проектування та практична реалізація розв’язку поставленої задачі. Тестування програми і результати її виконання. Висновки.
5. Перелiк графiчного матерiалу (з точним зазначенням обов`язкових креслень) Таблиці та схеми представлено в роботі. Презентація додається.
6. Дата видачі завдання «15» листопада 2019 р.

**АНОТАЦІЯ**

В курсовії роботі розглядаеться поняття, положення, задачі. Використання бази даних та використання класів. Розповідаеться про дизайн квартир та говориться про іх види.

Практична робота виконана у програмі Microsoft Visual Studio 2017, в середовищі розробки C# та SFML.

Курсова робота містить 26 сторінок, 4 розділи, 2 малюнки.

**Зміст**

[**ВСТУП** 6](#_Toc532674954)

[**1.ЗАГАЛЬНІ ВІДОМОСТІ ПРО ПРЕДМЕТНУ ОБЛАСТЬ** 7](#_Toc532674955)

[**1.1 С++** 7](#_Toc532674956)

[**1.2 Класи в С++** 8](#_Toc532674957)

[**2.КЛАСИФІКАЦІЯ БАЗ ДАНИХ** 10](#_Toc532674958)

[**2.1 Область застосування баз даних** 12](#_Toc532674959)

[**3.Основні відомості про середовище розробки** 13](#_Toc532674959)

[**3.1Microsoft Visual Studio** 13](#_Toc532674960)

[**3.2 Microsoft SQL Server** 14](#_Toc532674965)

[**3.3 Структура таблиці** 16](#_Toc532674968)

[**4. ОПИС ДИЗАЙНУ КВАРТИР** 18](#_Toc532674969)

[**5. ОХОРОНА ПРАЦІ** 21](#_Toc532674969)

[**ВИСНОВОК** 24](#_Toc532674970)

[**Список використаних джерел** 25](#_Toc532674971)

Додатки

Додаток А. Блок-схема програмного продукту.

Додаток В. Лістінг програмного продукту.

Додаток С. Скрін-шоти функціонування програмного продукту.

**ВСТУП**

**Microsoft Visual Studio** — серія продуктів фірми Microsoft, які включають [інтегроване середовище розробки](https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D1%81%D0%B5%D1%80%D0%B5%D0%B4%D0%BE%D0%B2%D0%B8%D1%89%D0%B5_%D1%80%D0%BE%D0%B7%D1%80%D0%BE%D0%B1%D0%BA%D0%B8) програмного забезпечення та ряд інших інструментальних засобів. Ці продукти дозволяють розробляти як [консольні програми](https://uk.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BA%D1%81%D1%82%D0%BE%D0%B2%D0%B8%D0%B9_%D1%96%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BA%D0%BE%D1%80%D0%B8%D1%81%D1%82%D1%83%D0%B2%D0%B0%D1%87%D0%B0), так і програми з [графічним інтерфейсом](https://uk.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84%D1%96%D1%87%D0%BD%D0%B8%D0%B9_%D1%96%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BA%D0%BE%D1%80%D0%B8%D1%81%D1%82%D1%83%D0%B2%D0%B0%D1%87%D0%B0), в тому числі з підтримкою технології [Windows Forms](https://uk.wikipedia.org/wiki/Windows_Forms), а також [веб-сайти](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B0%D0%B9%D1%82), [веб-застосунки](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%B7%D0%B0%D1%81%D1%82%D0%BE%D1%81%D1%83%D0%BD%D0%BE%D0%BA), [веб-служби](https://uk.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%BB%D1%83%D0%B6%D0%B1%D0%B0) як в [рідному](https://uk.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D0%B8%D0%B9_%D0%BA%D0%BE%D0%B4), так і в [керованому](https://uk.wikipedia.org/wiki/%D0%9A%D0%B5%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B9_%D0%BA%D0%BE%D0%B4) кодах для всіх платформ, що підтримуються [Microsoft Windows](https://uk.wikipedia.org/wiki/Microsoft_Windows), [Windows Mobile](https://uk.wikipedia.org/wiki/Windows_Mobile), [Windows Phone](https://uk.wikipedia.org/wiki/Windows_Phone_8), [Windows CE](https://uk.wikipedia.org/wiki/Windows_CE), [NET Framework](https://uk.wikipedia.org/wiki/.NET_Framework), [NET Compact Framework](https://uk.wikipedia.org/wiki/.NET_Compact_Framework) та [Microsoft Silverlight](https://uk.wikipedia.org/wiki/Microsoft_Silverlight).

**Об'єктно - орієнтоване програмування (ООП)** – це модель програмування яка базується на стверджені того, що програма це сукупність об’єктів які взаємодіють між собою. Кожен об’єкт в цій моделі є незалежним, і він здатний отримувати, обробляти дані та відправляти ці дані іншим об’єктам. В ООП використано моделі успадкування, модульності, поліморфізму та інкапсуляції.

Основним поняттям ООП є об'єкт. **Об'єкт** можна визначити як певну сукупність даних(характеристик об'єкта) та методів роботи з ними. Для класифікації об'єктів у ООП використовують класи. **Клас** служить зразком для створення обєкту, тобто об'єкт є нічим іншим, ніж копією класу.

1. **Загальні відомості про предметну область**

# 1.1 С++

[**C**](http://www.znannya.org/?view=concept:212)**++** ­­­-універсальна мова програмування високого рівня з підтримкою декількох парадигм програмування. Зокрема: об’єктно-орієнтованої та процедурної. Розроблена Б’ярном Страуструпом (англ. Bjarne Stroustrup) в AT&T Bell Laboratories (Мюррей-Хілл, Нью-Джерсі) у 1979 році та названа «С з класами». Страуструп перейменував мову у [C](http://www.znannya.org/?view=concept:212)++ у 1983 р. Базується на мові Сі. Визначена стандартом ISO/IEC 14882:2003.

У 1990-х роках С++ стала однією з найуживаніших мов програмування загального призначення.

При створенні С++ прагнули зберегти сумісність з мовою С. Більшість програм на С справно працюватимуть і з компілятором С++. С++ має синтаксис, заснований на синтаксисі С.

Нововведеннями С++ порівняно з С є:

* підтримка об'єктно-орієнтованого програмування через класи;
* підтримка узагальненого програмування через шаблони;
* доповнення до стандартної бібліотеки;
* додаткові типи даних;
* обробка винятків;
* простори імен;
* вбудовані функції;
* перевантаження операторів;
* перевантаження імен функцій;
* посилання і оператори управління вільно розподіленою пам'яттю.

# 1.2 Класи в С++

Класи і об'єкти в С є основними концепціями об'єктно-орієнтованого програмування - ООП. Об'єктно-орієнтоване програмування – розширення структурного програмування, в якому основними концепціями є поняття класів і об'єктів. Основна відмінність мови програмування С++ від С полягає в тому, що в С немає класів, а отже мова С не підтримує ООП, на відміну від С++ .

Щоб зрозуміти, для чого ж насправді потрібні класи, проведемо аналогію з яким-небудь об'єктом з повсякденного життя, наприклад, з велосипедом. Велосипед – це об'єкт, який був побудований згідно з кресленнями. Так ось, ці самі креслення грають роль класів в ООП. Таким чином класи – це деякі описи, схеми, креслення по яких створюються об'єкти. Тепер зрозуміло, що для створення об'єкта в ООП необхідно спочатку скласти креслення, тобто класи. Класи мають свої функції, які називаються методами класу. Пересування велосипеда здійснюється за рахунок обертання педалей, якщо розглядати велосипед з точки зору ООП, то механізм обертання педалей – це метод класу. Кожен велосипед має свій колір, вага, різні складові - все це властивості. Причому у кожного створеного об'єкта властивості можуть відрізнятися. Маючи один клас, можна створити необмежено кількість об'єктів (велосипедів), кожен з яких буде володіти однаковим набором методів, при цьому можна не замислюватися про внутрішню реалізації механізму обертання педалей, коліс, спрацьовування системи гальмування, так як все це вже буде визначено в класі. Розібравшись з призначенням класу, дамо йому грамотне визначення.

Класи в С**++** - це абстракція, що описує методи, властивості, ще не існуючих об'єктів. Об'єкти - конкретне уявлення абстракції, що має свої властивості і методи. Створені об'єкти на основі одного класу називаються екземплярами цього класу. Ці об'єкти можуть мати різну поведінку, властивості, але все одно будуть об'єктами одного класу. В ООП існує три основних принципи побудови класів:

**Інкапсуляція** - це властивість, що дозволяє об'єднати в класі і дані, і методи, які працюють з ними і приховати деталі реалізації від користувача.

Досягається шляхом поєднання в одному записі мови програмування структур даних з процедурами і функціями, які маніпулюють полями даних цього запису, для отримання нового типу даних - класу. Інкапсуляція дозволяє захистити по інтерфейсу доступ до полів і методів. Доступ дозволяється лише до відкритих методів та полів. Повна сукупність методів і тонкощі їхньої реалізацій є прихованими.

**Спадкування** - це властивість, що дозволяє створити новий клас-нащадок на основі вже існуючого, при цьому всі характеристики класу батька присвоюються класу-нащадка.

Число абстракцій в складних програмних системах набагато перевищує наші можливості їхнього усвідомлення. Інкапсуляція частково допомагає усунути цю перешкоду, прибираючи з поля зору внутрішній зміст абстракцій.

Однак значне спрощення розуміння складних завдань досягається за рахунок ускладнення ієрархії. Під ієрархією тут розуміється впорядкування абстракцій, розташування їх за рівнями. Ускладнення ієрархії від рівня до рівня досягається за рахунок наслідування.

**Поліморфізм** - властивість класів, що дозволяє використовувати об'єкти класів з однаковим інтерфейсом без інформації про тип і внутрішню структуру об'єкта.

Кожна властивість побудови класів ми розглянемо докладно в міру необхідності, а поки просто запам'ятайте ці три. А тепер повернемося до класів, для початку розглянемо структуру оголошення класів.

Метою поліморфізму є використання одного імені для завдання загальних для класу дій, причому кожен об'єкт або клас ієрархії має можливість по-своєму реалізувати цю дію своїм власним, відповідним для нього, кодом. Таким чином, поліморфізм є властивістю класів розв'язувати схожі за змістом проблеми різними способами.

[**2. КЛАСИФІКАЦІЯ БАЗ ДАНИХ**](#_Toc532674957)

Серед користувачів БД виділяють такі категорії:

**Адміністратор бази даних (АБД)** - ідеолог і головний конструктор бази даних. Ніхто, крім адміністратора, не має права вносити зміни до структури БД, тобто, змінювати назви полів, видаляти або вводити нові поля.

**Системні і прикладні програмісти,** При експлуатації великих за розмірами БД у АБД може бути певний штат таких працівників. Головним завданням **системних програмістів** є контроль за функціонуванням банку даних, а також розробка програм, що розширюють можливості СУБД. **Прикладні програмісти** розробляють програми обробки даних, що містяться в БД, відповідно до потреб, що виникають. (При експлуатації простих баз даних функції системних і прикладних програмістів, а також функції АБД, реалізуються однією людиною).

**Широке коло користувачів (кінцеві користувачі)** - це фахівці, які використовують відомості, що містяться в БД, у своїй повсякденній роботі. Саме для цієї категорії і розробляються бази даних.

Бази даних підрозділяються на:

1. централізовані;

2. розподілені.

**Централізована база даних** зберігається в пам’яті однієї обчислювальної системи. Якщо ця обчислювальна система є компонентом мережі ЕОМ, можливий розподілений доступ до такої бази. Такий спосіб використання баз даних часто застосовують в локальних мережах ПК.

**Розподілена база даних** складається з декількох, можливо пересічних або навіть дублюючих один одного частин, що зберігаються в різних ЕОМ обчислювальної мережі. Робота з такою базою здійснюється за допомогою системи управління розподіленою базою даних (СУРБД). (Такі БД ми далі розглядати не будемо).

Бази даних розділяються на:

1. бази даних з локальним доступом;

2. бази даних з віддаленим (мереженим) доступом.

Системи централізованих баз даних з мереженим доступом припускають різну архітектуру подібних систем:

1. файл-сервер;

2. клієнт-сервер.

**Файл-сервер,** архітектура систем БД з мереженим доступом припускає виділення однієї з машин мережі як центральної (сервер файлів). На такій машині зберігається спільно використовувана централізована БД. Всі інші машини мережі виконують функції робочих станцій, за допомогою яких підтримується доступ призначеної для користувача системи до централізованої бази даних. Файли бази даних відповідно до призначених для користувача запитів передаються на робочі станції, де в основному і проводиться обробка. При великій інтенсивності доступу до одних і тих же даних продуктивність інформаційної системи падає. Користувачі можуть створювати також на робочих станціях локальні БД, які використовуються ними монопольно.

**Клієнт-сервер,** у цій концепції мається на увазі, що крім зберігання централізованої бази даних, центральна машина (сервер бази даних) повинна забезпечувати виконання основного об’єму обробки даних. Запит на дані, видаваний клієнтом (робочою станцією), породжує пошук і витягання даних на сервері. Дані (але не файли), що витягуються , транспортуються по мережі від сервера до клієнта. Специфікою архітектури клієнт-сервер є використання мови запитів SQL.

**2.1 Області застосування баз даних**

Однією з важливих сфер комп'ютеризації суспільства є автоматизація управлінсько-виробничої, офісної та іншої засновницької роботи, де необхідні збір, зберігання, переробка, отримання, передача інформації, допомога в прийнятті рішень.

Автоматизація на основі персональних комп'ютерів ПК змінює стереотипи обробки інформації, дозволяє організувати функціонування підприємств і організацій на базі нової інформаційної технології.  
 Використання ПК як інструмент обробки даних в різних областях людської діяльності підвищує інформаційну культуру суспільства, сприяє безболісного переходу до інформаційного суспільства, де інформація є найважливішим стратегічним сировиною нарівні з матеріальними, енергетичними, людськими та ін. Ресурсами.

Від вчасно отриманої, правильно обробленої і чітко представленої інформації часто залежить ефективність прийнятих рішень і їх результат.  
 Якщо постаратися класифікувати існуючі області застосування баз даних, а так само оцінити перспективи їх розвитку в даний час, то можна отримати приблизний список найбільш поширених класів, які отримали поширення і застосування у всіх областях застосування баз даних.

Цей список буде виглядати наступним чином:  
 1. Документографичні та документальні застосовуються у всіх базах органів влади і управління  
 2. Бази даних з промислової, будівельної та сільськогосподарської продукції  
 3. Бази даних по економічній і кон'юнктурної інформації статистична, кредитно-фінансова, зовнішньоторговельна  
 4. Фактографічні бази соціальних даних, що включають відомості про населення і про соціальному середовищі  
 5. Бази даних транспортних систем  
 6. Довідкові дані для населення та установ енциклопедії та довідники, розклади літаків і поїздів, адреси та телефони громадян і організацій  
 7. Ресурсні бази даних, що включають фактографічну інформацію про природні ресурси земля, вода, надра, біоресурси, гідрометеорологія, вторинні ресурси і відходи, екологічна обстановка  
 8. Фактографічні бази і банки наукових даних, що забезпечують фундаментальні наукові дослідження  
 9. Фактографічні бази даних в області культури і мистецтва  
 10. Лінгвістичні бази даних, тобто машинні словники різного типу і призначення.

**3. ОСНОВНІ ВІДОМОСТІ ПРО СЕРЕДОВИЩЕ РОЗРОБКИ**

**3.1 Microsoft Visual Studio**

Microsoft Visual Studio – інтегроване середовище розробки програмного забезпечення від фірми Microsoft. Дане середовище дозволяє створювати різноманітні програмні продукти: консольні програми, програми з графічним інтерфейсом, наприклад віконні додатки Windows Forms, а також Web-додатки тощо.

Середовище Visual Studio дозволяє розробляти додатки, використовуючи різні мови програмування: Visual C#, Visual Basic, Visual F#, Visual C++, Python і т.д. (рис. 1). Також існує можливість розробляти додатки не тільки під Windows, а і під інші популярні платформи: Android, iOS.

Версія Visual Studio Community є абсолютно безкоштовною для учнів, студентів та розробників програм з відкритим програмним кодом.

 MS Visual Studio кожне окреме застосування є рішенням (solution), що складається з одного чи декількох проектів (project). Одночасно можна відкрити тільки одне рішення (з розширенням. sln), при роботі над кількома рішеннями одночасно потрібно запускати декілька вікон Visual Studio.

Visual Studio надає шаблони для проектів найбільш поширених типів. Використання проектів і їх шаблонів дозволяє користувачеві зосередитися на реалізації окремої функції, в той час як проект буде виконувати загальне управління та завдання побудови.

Для створення нового проекту використовується майстер застосувань. Майстер створення програм надає користувальницький інтерфейс для створення проекту за шаблоном та створення шаблонів для файлів вихідних текстів. Майстер налаштовує структуру програми, основні меню і панелі інструментів, забезпечує включення деяких заголовних файлів.

Елементи графічного інтерфейсу MS Visual Studio характерні для Windows-програм. Розмір і форма вікон визначається конкретною конфігурацією системи. Користувач має можливість змінювати розмір і розташування окремих елементів, згортати їх з тим, щоб збільшити місце для інших, необхідних в даний момент, елементів.

У головному вікні Visual Studio можна виділити декілька основних елементів:

1. меню та набір інструментальних панелей, де зосереджені команди для роботи в IDE;
2. вікно Провідник рішень (Solution Explorer), що дозволяє переглядати склад проектів, що входять у рішення, у вигляді ієрархічної структури, а також зв'язки між проектами та їх компонентами;
3. вікно редактора, що служить для набору тексту програми і підтримує автодоповнення та підсвітку ситнаксису;
4. вікно виведення стану Output, в якому відображається інформація про хід побудови (збірки) програми та виявлені помилки.

**3.2 Microsoft SQL Server**

Microsoft SQL Server - система керування базами даних, розроблена корпорацією Microsoft і працює тільки під Windows.

SQL Server використовує процедурне розширення T-SQL (Transact-SQL). Сама мова SQL цієї СУБД відрізняється від інших реалізацій SQL і іноді розглядається як діалект T-SQL.

T-SQL підтримує програмні блоки (у вигляді процедур, які зберігаються в базі даних в скомпільованому вигляді і можуть бути виконані пізніше створення, або анонімних блоків, які не можуть бути скомпільовані і використовуються тільки з вигляді скриптів), оператори управління потоком обчислень (в т .ч. умовні переходи і цикли) і локальні змінні. Крім того, T-SQL надає ряд функцій для роботи з рядками і датами, математичних функцій і т.д.

Для адміністрування БД і розробки додатків для СУБД SQL Server може використовуватися ряд інструментів, що надаються корпорацією Microsoft:

1. SQLCMD - інструмент з інтерфейсом типу командного рядка, призначений для виконання команд SQL інтерактивно або з скрипта; використовується в якості інсталяційного інтерфейсу за замовчуванням.
2. Microsoft Visual Studio включає підтримку розробки для SQL Server.
3. SQL Server Management Studio - IDE для конфігурації і управління БД SQL Server.

**Функціональність**

SQL Server підтримує дзеркалювання та кластеризацію баз даних. Кластер серверу SQL-це сукупність однаково конфігурованих серверів; така схема допомагає розподілити робоче навантаження між декількома серверами.

Усі сервери мають одне віртуальне ім'я, а дані розподіляються за IP-адресами машин кластеру протягом робочого циклу. Також у разі відмови або збою на одному з серверів кластеру доступне автоматичне перенесення навантаження на інший сервер.

SQL Server підтримує надлишкове дублювання даних за трьома сценаріями:

1. Знімок: Виконується «знімок» бази даних, який сервер відправляє одержувачам.
2. Історія змін: Всі зміни бази даних безперервно передаються користувачам.
3. [Синхронізація](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BD%D1%85%D1%80%D0%BE%D0%BD%D1%96%D0%B7%D0%B0%D1%86%D1%96%D1%8F_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) з іншими серверами: Бази даних декількох серверів синхронізуються між собою. Зміни усіх баз даних відбуваються незалежно на кожному сервері, а під час синхронізації відбувається звірка даних. Дублювання такого типу передбачає можливість вирішення протиріч між базами даних.

SQL Server 2005 має вбудовану підтримку .NET Framework. Завдяки цьому, процедури бази даних, що зберігаються, можуть бути написані на будь-якій мові платформи .NET з використанням повного набору бібліотек, доступних для .NET Framework. На відміну від інших процесів, .NET Framework виділяє додаткову пам'ять і будує засоби керування SQL Server, не використовуючи вбудовані засоби Windows. Це підвищує продуктивність порівняно із загальними алгоритмами Windows, оскільки алгоритми розподілу ресурсів спеціально налагоджені для використання у структурах SQL Server.

**3.3 Структура таблиці**

В таблиця Products зберігаються такі стовбці як:

* 1. Id - тип INT, є первинним ключем;
  2. Design – тип NCHAR(50), може бути NULL;
  3. Surname – тип NCHAR(50), може бути NULL;
  4. Name – тип NCHAR(50), може бути NULL;
  5. Patronymic – тип NCHAR(50), може бути NULL;
  6. Age – тип NCHAR(50), може бути NULL;
  7. ManWoman – тип NCHAR(50), може бути NULL;

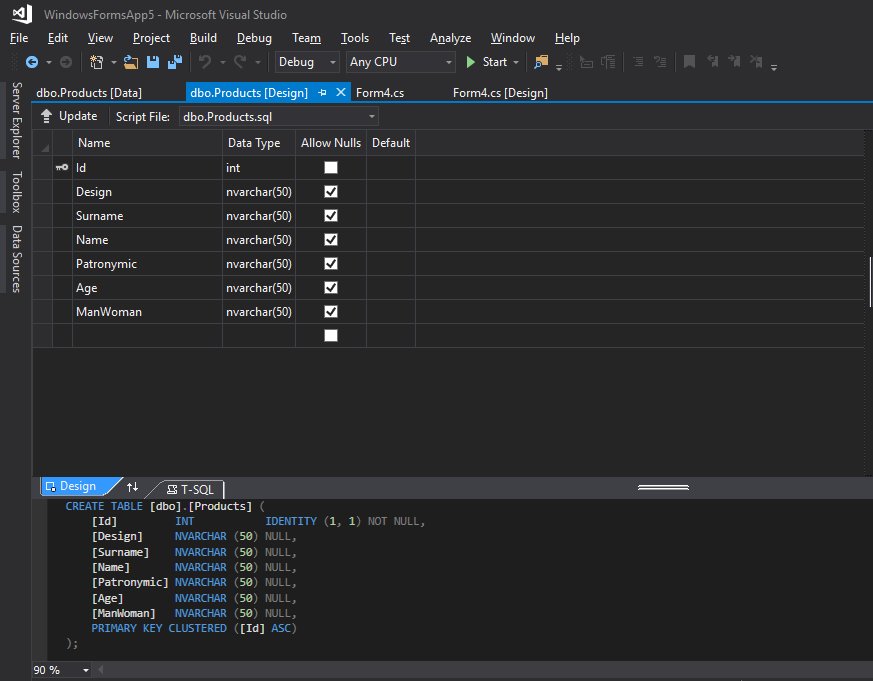


Рис 1. Таблиця Products

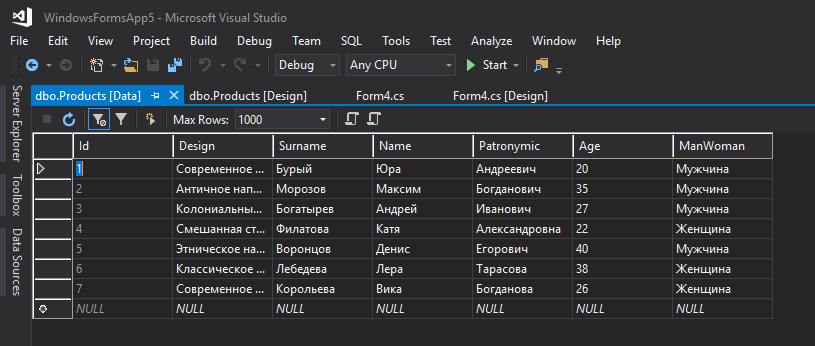


Рис 2.Інформація про дизайн квартир

**4. Опис дизайну квартир**

Зазвичай перед початком ремонту власник житла переглядає багато інформації в інтернеті, щоб визначитися, який стиль інтер'єру йому подобається. Оскільки вибір оздоблення, меблів та декору зростає, визначитися та обрати правильні стилі інтер'єру для оформлення стає все важче.

В загальному можна виділити три напрямки стилів - класичні, сучасні та етнічні стилі інтер'єру. Часто буває так, що в приміщенні використовують змішаний стиль, але для цього потрібна кваліфікація та високий художній смак. Це під силу тільки досвідченим дизайнерам.

### Сучасна класика

Сучасна класика - один з найпоширеніших стилів інтер'єру. Цей стиль не прив'язаний до модних тенденцій, які постійно змінюються. Гармонія та елегантність, які є характерними рисами сучасної класики постійно залишаються актуальними.  
 Дизайнерські рішення цього стилю бездоганно підходять не тільки для оформлення житлових приміщень, але й для офісів, ресторанів та бутиків. Меблі в стилі сучасна класика виглядають монументально та дорого і являються головним елементом в створенні інтер'єру.  
 З матеріалів переважають натуральні камінь та дерево. Дизайн інтер'єру часто доповнюють такими елементами як колони, ліпнина та багети. В наші дні цей стиль часто поєднують з мінімалізмом, приміняючи мінімум декоративних елементів.

### Арт-Деко

Арт-деко - стиль в дизайні, архітектурі та мистецтві, який з'явився в 30-х роках ХХ століття в наслідок змішування класичних, західних стилів зі стилями країн сходу та інших екзотичних країн. В ньому зустрічаються африканські, египетські, індійські та азіатські мотиви.  
Арт-Деко відрізняється різноманітністю та прагне задовільнити індивідуальний смак вибагливого замовника. В меблях та декорі використовуються дорогі та екзотичні матеріали - рідкісні породи дерева, слонова кістка, шкіри екзотичних тварин. Дороге скло та кристали, масивні люстри та різноманітність освітлення - все це можна зустріти в стилі Арт-Деко.

### Функціоналізм

Функціоналізм виник в наслідок індустріалізації в 20-х роках в Німеччині. Основою цього стилю - є зручність, функціональність та раціональне використання простору, тому він ідеально підходить для невеликих квартир.

. Обклеївши стіни шпалерами світлого кольору, ви отримаєте відчуття свободи та незавантаженості простору. На вікнах акуратні жалюзі, але можливі і прямі однотонні штори без всяких ламбрекенів і інших прикрас. Декорування зводиться до мінімуму, або відсутнє зовсім. Головне пам'ятати, що переваги стилю функціоналізм - це прямолінійність, чистота форм і кольорів, відсутність надлишків і індустріальний шарм

**Мінімалізм**

Мінімалізм - стиль гранично лаконічних форм, який характеризує повна відсутність декору, орнаментів або деталізації фасадної обробки. Предмети і простір втрачають форми, межі, створюється ілюзія нескінченного простору і невагомості.Для сучасного Мінімалізму характерно грамотне моделювання житлового простору з набором тільки необхідних предметів, які розкривають композиційний задум, і спокійного, розсіяного світла. Приміщення зоновано по можливості без внутрішніх перегородок, з великими вікнами і багаторівневим освітленням. Вони настільки насичені світлом, що навколишній ландшафт, зливаючись з житловим простором, стає частиною його інтер'єру.

У колірній гамі переважає білий, сірий і чорний тон. А блиск скла, хромованого металу, колір природної деревини або цегли доповнюють палітру.

### Лофт

### Лофт нестандартний стиль, що прийшов до нас зі США. Зазвичай так називається горище або приміщення промислового зразка, переобладнане під житло. Нерідко стиль Лофт реалізується на верхніх поверхах занедбаних фабрик і будинків.

### Головна вимога до приміщення - воно повинно бути просторим, тобто більше 100 кв. м. Простір інтер'єру в стилі лофт максимально відкритий, не перевантажений перегородками і міжкімнатними стінами. Зональне ділення за функціями зазвичай є досить умовним і роль поділу відводиться дизайну, а не конструкціям. Особливу цінність лофту надають деталі, успадковані з минулого життя будівлі: труби, вентиляція не приховані, стіни без обробки з грубої цегли.

### На даний час стиль Лофт став дуже популярним, особливо в інтер'єрах громадських приміщень (ресторани, нічні клуби, виставкові зали та галереї).

### Східний

### Східний інтер'єр поєднує в собі стилістику країн Сходу, яка передбачає яскраві соковиті кольори: рожевий, блакитний, жовтий на тлі чорного або білого, які можуть бути пом'якшені темними, або світлими породами дерева. Так само задають східний відтінок інтер'єру багаті малюнки або хитромудрі геометричні фігури. І, звичайно, тканини в "східному стилі" - блискучий шовк і муар, повітряна органза, переливаючись чарівним сяйвом, нагадують палаци магараджів і чарівні казки Шахерезади.Бароко

Одна з характерних рис бароко - просторові ілюзії, які супроводжують інтер'єри цього стилю. Компоненти, традиційні для барочних інтер'єрів - дзеркала, стелі, стіни, екрани - розписані зображеннями, засновані на картинах життя небожителів. На стінах висять гобелени, що містять зображення пейзажів, такі гобелени можна побачити і в сучасних багатих будинках. Урочистий пафос інтер'єру посилюється за допомогою великої кількості важких тканин, килимів, багатих виблискуючих люстр, позолоченої ліпнини, в якій переважає тематика зображень листя і плодів. Криволінійні, округлі контури надають декору динаміки і мальовничості.

### Романтизм

Романтизм зародився також у Франції наприкінці 18 століття. На початку віку технічного прогресу, багато визначних діячів мистецтв задумуються про те, що в бажанні встигати за новими винаходами та технологіями, людина забуває про зв'язок з природою та все менше розуміє власні почуття.  
 Ніжні текстури та палітра кольорів, плавні лінії, вишукані та легкі конструкції меблів, натуральні матеріали в декорі та оздобленні, заставляють уповільнити ритм життя та прислухатись до власних почуттів.

Розкіш та шик не є характерними ознаками романтизму, та все ж важко назвати цей стиль економним, або недорогим. Головна мета даного стилю - створити комфортну, затишну та природну атмосферу в домівці.

**5.ОХОРОНА ПРАЦІ**

Будівлі та приміщення, де розміщені робочі місця, повинні відповідати вимогам нормативно-технічної та експлуатаційної документації виробника персональних комп’ютерів ДСанПіН 3.3.2-007-98 та Правил. Будівлі та приміщення, де розміщені робочі місця операторів, мають бути не нижче другого ступеня вогнестійкості. Для всіх будівель і приміщень, де знаходяться робочі місця, повинно бути визначено клас зони згідно з НПАОП 40.1-1.01-97. Відповідне позначення повинно бути нанесено на вхідних дверях кожного приміщення. Не дозволяється розташування приміщень з робочими місцями у підвалах і цокольних поверхах. Неприпустимим є розташування приміщень категорій А і Б, а також виробництв з мокрими технологічними процесами поряд з приміщеннями, де розташовуються робочі місця, а також над ними чи під ними. При цьому площа приміщення має бути не менше 6,0 кв. м. із розрахунку на одне робоче місце, а об’єм – не менше 20,0 куб. м.

Віконні прорізи приміщень для роботи з персональними комп’ютерами мають бути обладнані регульованими пристроями (жалюзі, завіски, зовнішні козирки. Для внутрішнього оздоблення приміщень з персональними комп‘ютерами слід використовувати дифузно-відбивні матеріали з коефіцієнтами відбиття для стелі 0,7-0,8, для стін 0,5-0,6. Покриття підлоги повинне бути матовим з коефіцієнтом відбиття 0,3-0,5. Поверхня підлоги має бути рівною, неслизькою, з антистатичними властивостями. Забороняється для оздоблення інтер’єру приміщень з персональними комп‘ютерами застосовувати полімерні матеріали (деревинно-стружкові плити, шпалери, що миються, рулонні синтетичні матеріали, шаруватий паперовий пластик тощо), що виділяють у повітря шкідливі хімічні речовини. Полімерні матеріали для внутрішнього оздоблення приміщень з персональними комп’ютерами можуть бути використані при наявності дозволу органів та установ державної санітарно-епідеміологічної служби. Приміщення можуть обладнуватись шафами для зберігання документів, магнітних дисків, полицями, стелажами, тумбами тощо з урахуванням вимог до площі приміщень.

**Організація та обладнання робочого місця**

При розміщенні робочих столів з персональними комп’ютерами слід дотримувати:

1. відстань між бічними поверхнями персональних комп’ютерів  1,2 м.;
2. відстань від тильної поверхні одного персонального комп’ютера до екрана іншого – 2,5 м.

За потреби особливої концентрації уваги під час виконання робіт суміжні робочі місця операторів необхідно відділяти одне від одного перегородками висотою 1,5 – 2м.

Конструкція робочого місця користувача персонального комп’ютера має забезпечити підтримання оптимальної робочої пози офісного працівника. Конструкція робочого столу має відповідати сучасним вимогам ергономіки і забезпечувати оптимальне розміщення на робочій поверхні використовуваного обладнання (дисплея, клавіатури, принтера) і документів. Висота робочої поверхні робочого столу має регулюватися в межах 680-800 мм, а ширина і глибина – забезпечувати можливість виконання операцій у зоні досяжності моторного поля (рекомендовані розміри: 600-1400мм, глибина – 800-1000мм).

Робочі місця слід розташовувати відносно світових прорізів так, щоб природне світло падало переважно з лівого боку. Монітор має розташовуватися на оптимальній відстані від очей користувача, що становить 600-700мм, але не ближче ніж за 600мм з урахуванням розміру літерно-цифрових знаків і символів. Розташування пристрою введення – виведення інформації має забезпечувати добру видимість монітору, зручність ручного керування в зоні досяжності моторного поля і за висотою – 900-1300мм, за шириною 400-500мм. Під матричні принтери потрібно підкладати вібраційні килимки для гасіння вібрації та шуму.

Не допускається:

1. виконувати обслуговування, ремонт та налагодження персонального комп’ютеру та периферійних пристроїв безпосередньо на робочому місці оператора;
2. зберігати біля персонального комп’ютеру та периферійних пристроїв папір, будь-які носії інформації (диски, флешки тощо), запасні блоки, деталі тощо, якщо вони не використовуються для поточної роботи;
3. відключати захисні пристрої, самочинно проводити зміни у конструкції та складі персонального комп’ютеру та периферійних пристроїв або їх технічне налагодження;
4. працювати з персональним комп’ютером, у яких під час роботи з’являються нехарактерні сигнали, нестабільне зображення на моніторі тощо;
5. працювати з матричним принтером за відсутності вібраційного килимка та зі знятою (піднятою) верхньою кришкою.

# ВИСНОВОК

Під час цієї роботи -я застосував свої знання з об’єктно-орієнтованого програмування та знання з бази даних. При використанні бібліотеки SFML я навчився працювати з об’єктами, та підключенням бази даних для кращого подання інформації .

Дізнався що SFML- це бібліотека для подання мультимедійних даних. SFML забезпечує простий інтерфейс для розробки ігор та інших мультимедійних додатків. Складається вона з п'яти модулів: system, window, graphics, audio і network. Про особливості всіх цих компонент і буде розказано в уроках.

# Список використаних джерел

1. Ламзин С. А., Сурдин В. Г. [Протозвёзды. Где, как и из чего формируются звёзды](http://www.astronet.ru/db/msg/1190949/index.html). – М. : Наука, 1992.
2. Горев А., Ахаян Р., Макашарипов С. Эффективная работа с СУБД-СПб.: Питер, 1997.
3. Принципи ООП [Електронний ресурс]. – Режим доступу: <https://learn.ztu.edu.ua/mod/page/view.php?id=9974>
4. Дизайн [Електронний ресурс]. – Режим доступу: <https://mauris-design.in.ua/styli-interyeru/>
5. База даних [Електронний ресурс]. – Режим доступу: http://lib.mdpu.org.ua/e-book/vstup/L5.htm
6. Основні поняття баз даних [Електронний ресурс]. – Режим доступу: <https://ru.bmstu.wiki/Microsoft_Visual_Studio>
7. С++[Електронний ресурс]. – Режим доступу: <http://cppstudio.com/uk/cat/274/281/>
8. Хомоненко А.Д., Цыганков В.М., Мальцев М.Г. Базы данных: Учебник для высших учебных заведений/ Под ред. Проф. А.Д. Хомоненко. – СПб: КОРОНА принт, 2000.
9. Основи мови програмування [Електронний ресурс]. – Режим доступу: http://progopedia.ru/implementation/microsoft-sql-server/
10. Міессеров К. Г. Новий погляд на освіту Сонячної системи і еволюцію Всесвіту. М.: Изд-во «Машинобудування», 1993.
11. ООП в С++ [Електронний ресурс]. – Режим доступу: <http://programming.in.ua/programming/basisprogramming/25-oop.html>
12. Верхні планети [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Microsoft\_SQL\_Server
13. Основні поняття [Електронний ресурс]. – Режим доступу: <https://gc.ua/uk/oxorona-praci-v-ofisi-vimogi-do-robochogo-miscya-ofisnogo-pracivnika/>
14. ООП-[Електронний ресурс]. – Режим доступу: <https://studfile.net/preview/5851327/page:11/>
15. База данних [Електронний ресурс]. – Режим доступу: https://learn.ztu.edu.ua/mod/page/view.php?id=9974

**ДОДАТКИ**

**Додаток А.**

true

true

false

false

**Закриває інформаційне вікно**

**if**

**Показує інформацію користувачу**

**Зчитування інформації**

**while**

**Підключення до бази даних**

**Додаток В**

**Лістинг програми симулятора**.

**Bикористовується SELECT, UPDATE, DELETE**

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 WindowsFormsApp5

{

public partial class Form4 : Form

{

SqlConnection sqlConnection;

public Form4()

{

InitializeComponent();

}

private void button5\_Click(object sender, EventArgs e)

{

this.Hide();

Form2 Form2 = new Form2();

Form2.Show();

}

private async void Form4\_Load(object sender, EventArgs e)

{

this.productsTableAdapter.Fill(this.database2DataSet.Products);

string connectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=D:\WindowsFormsApp5\WindowsFormsApp5\Database2.mdf;Integrated Security=True";

sqlConnection = new SqlConnection(connectionString);

sqlConnection.Open();

SqlDataReader sqlReader = null;

SqlCommand command = new SqlCommand("SELECT \* FROM [Products]", sqlConnection);

try

{

sqlReader = command.ExecuteReader();

while (sqlReader.Read())

{

listBox1.Items.Add(Convert.ToString(sqlReader["id"]) + ")" + Convert.ToString(sqlReader["Design"]) + "," + Convert.ToString(sqlReader["Surname"]) + " " + Convert.ToString(sqlReader["Name"]) + " " + Convert.ToString(sqlReader["Patronymic"]) + "," + Convert.ToString(sqlReader["Age"]) + " лет," + Convert.ToString(sqlReader["ManWoman"]));

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(), MessageBoxButtons.OK, MessageBoxIcon.Error);

}

finally

{

if (sqlReader != null)

{

sqlReader.Close();

}

}

}

private void віходToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (sqlConnection != null && sqlConnection.State != ConnectionState.Closed)

sqlConnection.Close();

}

private void обновитьToolStripMenuItem\_Click(object sender, EventArgs e)

{

listBox1.Items.Clear();

SqlDataReader sqlReader = null;

SqlCommand command = new SqlCommand("SELECT \* FROM [Products]", sqlConnection);

try

{

sqlReader = command.ExecuteReader();

while (sqlReader.Read())

{

listBox1.Items.Add(Convert.ToString(sqlReader["id"]) + ")" + Convert.ToString(sqlReader["Design"]) + "," + Convert.ToString(sqlReader["Surname"]) + "," + Convert.ToString(sqlReader["Name"]) + "," + Convert.ToString(sqlReader["Patronymic"]) + "," + Convert.ToString(sqlReader["Age"]) + "," + Convert.ToString(sqlReader["ManWoman"]));

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString(), ex.Source.ToString(), MessageBoxButtons.OK, MessageBoxIcon.Error);

}

finally

{

if (sqlReader != null)

{

sqlReader.Close();

}

}

}

private async void Form4\_FormClosing(object sender, FormClosingEventArgs e)

{

if (sqlConnection != null && sqlConnection.State != ConnectionState.Closed)

sqlConnection.Close();

}

private async void button6\_Click(object sender, EventArgs e)

{

if (label17.Visible)

label17.Visible = false;

if (!string.IsNullOrEmpty(textBox6.Text) && !string.IsNullOrWhiteSpace(textBox6.Text)

&& !string.IsNullOrEmpty(textBox1.Text) && !string.IsNullOrWhiteSpace(textBox1.Text)

&& !string.IsNullOrEmpty(textBox2.Text) && !string.IsNullOrWhiteSpace(textBox2.Text)

&& !string.IsNullOrEmpty(textBox3.Text) && !string.IsNullOrWhiteSpace(textBox3.Text)

&& !string.IsNullOrEmpty(textBox4.Text) && !string.IsNullOrWhiteSpace(textBox4.Text)

&& !string.IsNullOrEmpty(textBox5.Text) && !string.IsNullOrWhiteSpace(textBox5.Text)

&& !string.IsNullOrEmpty(textBox7.Text) && !string.IsNullOrWhiteSpace(textBox7.Text))

{

SqlCommand command = new SqlCommand("UPDATE [Products] SET [Design] = @Design, [Surname] = @Surname, [Name] = @Name, [Patronymic] = @Patronymic, [Age] = @Age, [ManWoman] = @ManWoman WHERE [id] = @id", sqlConnection);

command.Parameters.AddWithValue("id", textBox6.Text);

command.Parameters.AddWithValue("Design", textBox1.Text);

command.Parameters.AddWithValue("Surname", textBox2.Text);

command.Parameters.AddWithValue("Name", textBox3.Text);

command.Parameters.AddWithValue("Patronymic", textBox4.Text);

command.Parameters.AddWithValue("Age", textBox5.Text);

command.Parameters.AddWithValue("ManWoman", textBox7.Text);

await command.ExecuteNonQueryAsync();

}

if (!string.IsNullOrEmpty(textBox6.Text) && !string.IsNullOrWhiteSpace(textBox6.Text))

{

}

else

{

label17.Visible = true;

label17.Text = "Все поля должны быть заполнены!!!";

}

}

private void button3\_Click(object sender, EventArgs e)

{

this.Hide();

Form2 Form2 = new Form2();

Form2.Show();

}

private void radioButton1\_CheckedChanged(object sender, EventArgs e)

{

if (radioButton1.Checked == true)

{

MessageBox.Show("Вы выбрали Мужчина");

textBox7.Text = System.Convert.ToString("Мужчина");

}

}

private void radioButton2\_CheckedChanged(object sender, EventArgs e)

{

if (radioButton2.Checked == true)

{

MessageBox.Show("Вы выбрали Женщина");

textBox7.Text = System.Convert.ToString("Женщина");

}

}

private void radioButton12\_CheckedChanged(object sender, EventArgs e)

{

if (radioButton12.Checked == true)

{

MessageBox.Show("Вы выбрали Античное направление");

textBox1.Text = System.Convert.ToString("Античное направление");

}

}

private void radioButton11\_CheckedChanged(object sender, EventArgs e)

{

if (radioButton11.Checked == true)

{

MessageBox.Show("Вы выбрали Классическое направление");

textBox1.Text = System.Convert.ToString("Классическое направление");

}

}

private void radioButton10\_CheckedChanged(object sender, EventArgs e)

{

if (radioButton10.Checked == true)

{

MessageBox.Show("Вы выбрали Этническое направление");

textBox1.Text = System.Convert.ToString("Этническое направление");

}

}

private void radioButton9\_CheckedChanged(object sender, EventArgs e)

{

if (radioButton9.Checked == true)

{

MessageBox.Show("Вы выбрали Современное направление");

textBox1.Text = System.Convert.ToString("Современное направление");

}

}

private void radioButton8\_CheckedChanged(object sender, EventArgs e)

{

if (radioButton8.Checked == true)

{

MessageBox.Show("Вы выбрали Колониальный стиль");

textBox1.Text = System.Convert.ToString("Колониальный стиль");

}

}

private void radioButton7\_CheckedChanged(object sender, EventArgs e)

{

if (radioButton7.Checked == true)

{

MessageBox.Show("Вы выбрали Смешанную стилистику");

textBox1.Text = System.Convert.ToString("Смешанную стилистику");

}

}

private async void button1\_Click(object sender, EventArgs e)

{

if (!string.IsNullOrEmpty(textBox1.Text) && !string.IsNullOrEmpty(textBox1.Text))

{

}

else

{

label17.Visible = true;

label17.Text = "login должн быть заполнен!";

}

if (label11.Visible)

{

label11.Visible = false;

}

if (!string.IsNullOrEmpty(textBox9.Text) && !string.IsNullOrWhiteSpace(textBox9.Text))

{

SqlCommand command = new SqlCommand("DELETE FROM [Products] WHERE [id] = @id", sqlConnection);

command.Parameters.AddWithValue("id", textBox9.Text);

await command.ExecuteNonQueryAsync();

}

else

{

label11.Visible = true;

label11.Text = "ID должен быть заполнен!!!";

}

}

private void productsBindingNavigatorSaveItem\_Click(object sender, EventArgs e)

{

this.Validate();

this.productsBindingSource.EndEdit();

this.tableAdapterManager.UpdateAll(this.database2DataSet);

}

}

}

**Bикористовується INSERT**

namespace WindowsFormsApp5

{

public partial class Form5 : Form

{

SqlConnection sqlConnection;

public Form5()

{

InitializeComponent();

}

private async void button2\_Click(object sender, EventArgs e)

{

if (label7.Visible)

{

label7.Visible = false;

}

SqlCommand command = new SqlCommand("INSERT INTO [Products] (Design, Surname, Name, Patronymic, Age, ManWoman) VALUES (@Design,@Surname,@Name,@Patronymic,@Age,@ManWoman)", sqlConnection);

if (!string.IsNullOrEmpty(textBox1.Text) && !string.IsNullOrWhiteSpace(textBox1.Text)

&& !string.IsNullOrEmpty(textBox2.Text) && !string.IsNullOrWhiteSpace(textBox2.Text)

&& !string.IsNullOrEmpty(textBox3.Text) && !string.IsNullOrWhiteSpace(textBox3.Text)

&& !string.IsNullOrEmpty(textBox4.Text) && !string.IsNullOrWhiteSpace(textBox4.Text)

&& !string.IsNullOrEmpty(textBox5.Text) && !string.IsNullOrWhiteSpace(textBox5.Text)

&& !string.IsNullOrEmpty(textBox6.Text) && !string.IsNullOrWhiteSpace(textBox6.Text))

{

command.Parameters.AddWithValue("Design", textBox1.Text);

command.Parameters.AddWithValue("Surname", textBox2.Text);

command.Parameters.AddWithValue("Name", textBox3.Text);

command.Parameters.AddWithValue("Patronymic", textBox4.Text);

command.Parameters.AddWithValue("Age", textBox5.Text);

command.Parameters.AddWithValue("ManWoman", textBox6.Text);

await command.ExecuteNonQueryAsync();

}

else

{

label7.Visible = true;

label7.Text = "Все поля должны быть заполнены!!!";

}

}

private async void Form5\_Load(object sender, EventArgs e)

{

string connectionString = @"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=D:\WindowsFormsApp5\WindowsFormsApp5\Database2.mdf;Integrated Security=True";

sqlConnection = new SqlConnection(connectionString);

await sqlConnection.OpenAsync();

}

private void radioButton1\_CheckedChanged\_1(object sender, EventArgs e)

{

if (radioButton1.Checked == true)

{

MessageBox.Show("Вы выбрали Античное направление");

textBox1.Text = System.Convert.ToString("Античное направление");

}

}

private void radioButton2\_CheckedChanged(object sender, EventArgs e)

{

if (radioButton2.Checked == true)

{

MessageBox.Show("Вы выбрали Классическое направление");

textBox1.Text = System.Convert.ToString("Классическое направление");

}

}

private void radioButton3\_CheckedChanged(object sender, EventArgs e)

{

if (radioButton3.Checked == true)

{

MessageBox.Show("Вы выбрали Этническое направление");

textBox1.Text = System.Convert.ToString("Этническое направление");

}

}

private void radioButton4\_CheckedChanged(object sender, EventArgs e)

{

if (radioButton4.Checked == true)

{

MessageBox.Show("Вы выбрали Современное направление");

textBox1.Text = System.Convert.ToString("Современное направление");

}

}

private void radioButton5\_CheckedChanged(object sender, EventArgs e)

{

if (radioButton5.Checked == true)

{

MessageBox.Show("Вы выбрали Колониальный стиль");

textBox1.Text = System.Convert.ToString("Колониальный стиль");

}

}

private void radioButton6\_CheckedChanged(object sender, EventArgs e)

{

if (radioButton6.Checked == true)

{

MessageBox.Show("Вы выбрали Смешанную стилистику");

textBox1.Text = System.Convert.ToString("Смешанную стилистику");

}

}

private void button4\_Click(object sender, EventArgs e)

{

Form3 Form3 = new Form3();

this.Hide();

Form3.ShowDialog();

}

private void radioButton7\_CheckedChanged(object sender, EventArgs e)

{

if (radioButton7.Checked == true)

{

MessageBox.Show("Вы выбрали Мужчина");

textBox6.Text = System.Convert.ToString("Мужчина");

}

}

private void radioButton8\_CheckedChanged(object sender, EventArgs e)

{

if (radioButton8.Checked == true)

{

MessageBox.Show("Вы выбрали Женщина");

textBox6.Text = System.Convert.ToString("Женщина");

}

}

private void label7\_Click(object sender, EventArgs e)

{

label7.Visible = true;

label7.Text = "Все поля должны быть заполнены!!!";

}

}

}

**Додаток С.**



Рис 1. Вхід, для входу використовується LOGIN та PASSWORD (admin, admin).



Рис 2. Тестовий вибір.



Рис 3. Вибір дизайну квартир.

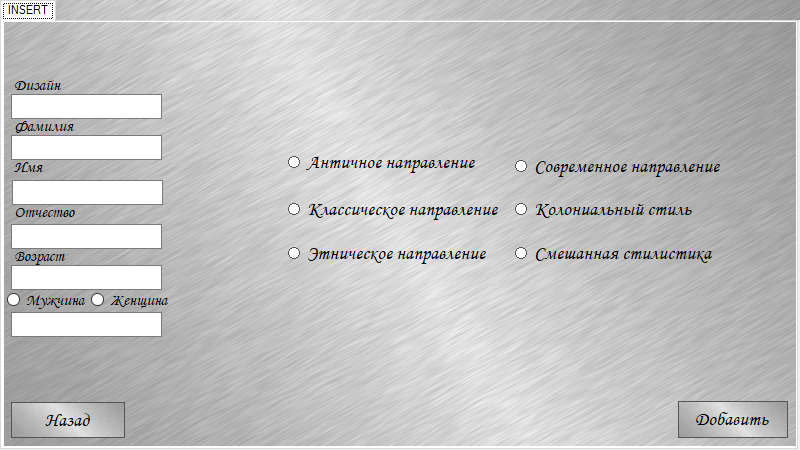


Рис 4. Тестовий приклад оформлення.

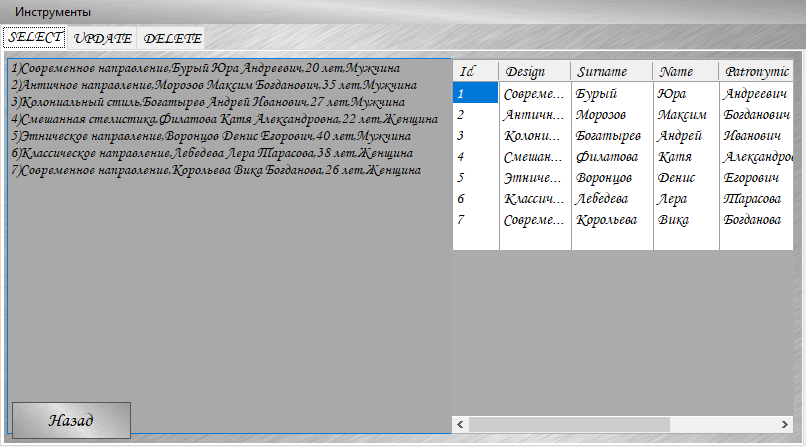


Рис 5. Тестовий приклад інформації з бази даних.