**Зміст**

[Перелік скорочень 4](#_Toc450815805)

[Вступ 5](#_Toc450815806)

[1. Характеристика предметної області і постановка задачі 6](#_Toc450815807)

[1.1. Аналіз предметної області 6](#_Toc450815808)

[1.2. Аналіз інформаційного забезпечення предметної області 9](#_Toc450815809)

[1.2.1. Аналіз існуючого програмного забезпечення предметної області 9](#_Toc450815810)

[1.2.2. Аналіз сучасних засобів створення програмного забезпечення 11](#_Toc450815811)

[1.3. Постановка задачі 13](#_Toc450815812)

[2. Проектування структури інформаційної системи 14](#_Toc450815813)

[2.1. Аналіз та автоматизація обробки інформаційних потоків 14](#_Toc450815814)

[2.2. Розробка структури інформаційної системи 21](#_Toc450815815)

[2.3. Вибір засобів розробки інформаційної системи 26](#_Toc450815816)

[2.3.1. Вибір мови програмування 26](#_Toc450815817)

[2.3.2. Вибір системи керування базами даних 27](#_Toc450815818)

[3. Програмна реалізація 33](#_Toc450815819)

[3.1. Структура і функціональне призначення модулів системи, їх взаємозв’язок 33](#_Toc450815820)

[3.2. Розробка програмних модулів 38](#_Toc450815821)

[3.2.1. Створення таблиць для збереження даних 38](#_Toc450815822)

[3.2.2. Проектування запитів для рішення задач 48](#_Toc450815823)

[3.2.3. Розробка інтерфейсу для введення і перегляду даних 50](#_Toc450815824)

[3.3. Тестування інформаційної системи 53](#_Toc450815825)

[3.4. Інструкція користувача 69](#_Toc450815826)

[3.5. Вимоги до програмних та технічних засобів 74](#_Toc450815827)

[Висновки 76](#_Toc450815828)

[Перелік посилань 77](#_Toc450815829)

Додатки

Перелік скорочень

|  |  |
| --- | --- |
| **Скорочення, термін, позначення** | **Пояснення** |
| БД | База даних |
| СКБД | Система керування базами даних |
| ПК | Персональний комп’ютер |
| АРМ | Автоматизоване робоче місце |
| MS | Microsoft |
| ПП | Програмний продукт |
| ІТ | Інформаційні технології |

Вступ

У наш час інформація оточує нас зі всіх сторін. У всіх установах та закладах зберігається велика кількість інформації. Для зручного її застосування використовують БД. Але не можна обмежитись тільки зберіганням інформації, для її обробки та застосування розробляються програмні продукти на основі СКБД. Тому перспективним є розвиток саме цієї ланки у розробці програмних продуктів.

Як було замічено вище, для роботи з даними використовують вiдповiднi програмні продукти. Такими продуктами є СКБД, які почали використовуватись після появи ПК. СКБД використовуються у всіх областях, починаючи від навчальних закладів завершуючи державними установами. Однією з переваг сучасних СКБД є низькі вимоги до апаратної частини ПК. Крім того кожна нова версія СКБД розробляється з думкою про користувача. Інтерфейс стає більш інтуітивним, а набір функцій розраховується таким чином, щоб пересічний користувач легко зміг зрозуміти суть.

Професія кухаря потребує використання великої кількості інформації та правильного її застосування. Зокрема, кухар відповідає за приготування страв, виконання ревізій, прийняття поставок. Тому для цього є доцільним застосування інформаційних технологій для автоматизації роботи кухаря [1].

1. Характеристика предметної області і постановка задачі

1.1. Аналіз предметної області

Підприємством громадського харчування є виробничо-торгова одиниця (їдальня, ресторан, кафе, закусочна, бар та ін.), яка виконує функції забезпечення населення харчуванням (виробництво кулінарної продукції, її реалізація й організація споживання), а також організовує цікаве дозвілля.

Процес перетворення сировини на готову продукцію є технологічним процесом. Він складається з послідовних стадій: приймання і зберігання сировини; обробка сировини і виробництво напівфабрикатів; виробництво готової продукції; реалізація готової продукції.

Залежно від організації технологічного процесу підприємства громадського харчування поділяють на ті, в яких технологічний процес здійснюється повністю, на заготівельні, доготівельні та на ті, що тільки реалізують готову продукцію [2].

Кухар є людиною, що професійно готує їжу для інших людей, як правило, в закладах громадського харчування.

Кухар готує супи, другі страви, іншу їжу, знає, як правильно зберігати продукти, готує різні страви за рецептами і вміє оформляти готові страви. Організовує зберігання продуктів відповідно до санітарно-гігієнічних норм. Попри те, що кухар на підприємствах громадського харчування користується встановленими рецептами страв, він може вносити до них зміни в залежності від якості сировини і контингенту споживачів. Працює в приміщенні, в умовах підвищеної температури.

Для роботи кухарем потрібна санітарна книжка.

Професію кухаря можна здобути в спеціальних коледжах, середньотехнічних навчальних закладах та професійно-технічних училищах.

Кухарів поділяють на наступні види:

* Начальник виробництва – до його відповідальності належить абсолютно все, що стосується роботи кухні, закладу і т. п., включаючи складання меню, підбір персоналу, економічні питання. Ця позиція вимагає навіть не стільки навичок приготування страв, скільки навичок менеджменту і управління. Це якраз та людина, яку європейці називають chef, head chef (але це не український «шеф-кухар»!).
* Шеф-кухар – це власне шеф-кухар, який відповідає за приготування страв на окремо взятому виробництві. Для європейської кухні, особливо невеликої, часто CDC і EC - одна і та ж особа. Слід зазначити, що CDC, як правило, відповідає лише за «свою» кухню, тоді як EC може відповідати, наприклад, відразу за всі аспекти кухні в кількох ресторанах власника. Іноді можна зустріти назву «Head of the Kitchen».
* Су-шеф – помічник і заступник шеф-кухаря. Також може відповідати за розпис робіт, внутрішню логістику; здатен підмінити шеф-кухаря за необхідності. Також може і допомагати іншим кухарям. Зрозуміло, що на великих виробництвах таких посад може бути навіть кілька.
* Кухар – відповідає за якісь виділені напрямки кулінарного виробництва.
* Кухар-кондитер – спеціалізується на [кондитерських виробах](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BD%D0%B4%D0%B8%D1%82%D0%B5%D1%80%D1%81%D1%8C%D0%BA%D1%96_%D0%B2%D0%B8%D1%80%D0%BE%D0%B1%D0%B8).
* Кухар-технолог – організовує процес приготування продуктів. Визначає якість сировини, розраховує його кількість для отримання порцій готових продуктів, калорійність добового раціону, складає меню та прейскуранти. Розподіляє обов'язки в бригаді кухарів. Контролює процес приготування кулінарної продукції, розробляє рецепти нових фірмових страв і складає на них технологічні карти. Оформлює необхідну документацію, інструктує кухарів. Веде повний облік матеріальних цінностей, обладнання, сировини, готової продукції [3].

Страва – поєднання продуктів харчування, які доведені до [кулінарної](https://uk.wikipedia.org/wiki/%D0%9A%D1%83%D0%BB%D1%96%D0%BD%D0%B0%D1%80%D1%96%D1%8F) готовності, порціоновані, оформлені і готові до споживання. Страви цінуються за їхні [смакові](https://uk.wikipedia.org/wiki/%D0%A1%D0%BC%D0%B0%D0%BA) властивості, які можна назвати найголовнішою ознакою всіх харчових продуктів, але смакове сприйняття дуже індивідуальне й важко піддається вербальному опису. Вживання страв у різних [народів](https://uk.wikipedia.org/wiki/%D0%9D%D0%B0%D1%80%D0%BE%D0%B4) особливе. Страви одного народу називають [кухнею](https://uk.wikipedia.org/wiki/%D0%9A%D1%83%D1%85%D0%BD%D1%8F) цього народу.

Можна умовно виділити такі аспекти, за якими будується словникове тлумачення назви страви:

1) рід страви (суп, кондитерський виріб, напій та ін.);

2) спосіб приготування (варіння, смаження);

3) склад компонентів (із чого готується);

4) зовнішній вигляд;

5) походження / культурний компонент (приналежність до національної кухні певного народу; не завжди вказується прямо, частіше виводиться з етимології слова);

6) типове споживання (холодним, гарячим, із чимось).

Страви традиційно поділяють згідно з етапами споживання їжі, на «перші страви», «другі страви» та «солодкі страви». Іншою класифікацією є перші страви, основна страва та десерт.

З точки зору організації процесу, спочатку збирається інформація про інгрідієнти, їх наявність за допомогою партій та поставок, далі кухар формує складові за допомогою інгрідієнтів, в свою чергу з складових по рецепту готуються страви.

Крім того кухарем контролюються ревізії (перевірки кількості використаних інгрідієнтів до кількість інгрідієнтів в наявності) та поставки.

Оскільки робота кухаря характерзується значним інформаційним навантаженням й повязана з необхідністю фіксувати приготовані страви ,ревізії та поставки, то є доцільною розробка програмного продукту, що дозволить автоматизувати виконання даних функцій.

Елементи предметної області

Отже предметну область харктеризують наступні параметри наведені в табл 1

1.2. Аналіз інформаційного забезпечення предметної області

1.2.1. Аналіз існуючого програмного забезпечення предметної області

Для автоматизації роботи кухаря на ринку програмних продуктів є декілька варіантів. Одне з рішень – використання офісного пакету Office від Microsoft, а саме: MS Word, MS Excel, MS Access [4].

MS Word дозволяє створювати таблиці, схеми, у рисунках зберігати рецепти страв. Але середа застосування MS Office трохи відрізняється від потреб кухаря, тому останній може мати труднощі з використанням. Кількість часу витраченого на виконання однієї операції не пропорційна результату, тому доцільніше використовувати Excel [5].

Excel – програмний продукт для роботи з електронними таблицями. При роботі кухаря з даним програмним продуктом, можуть виникнути проблеми з наповненням таблиць, адже як ми знаємо у Excel немає зв’язків між   
таблицями [6].

ACcess

Хоча використання офісного пакету від компанії Microsoft можливе, оптимальнішим варіантом буде застосування спеціальних програмних продуктів.

Одним з рішень є програмний продукт «iОбщепит» від компанії 1С [7]. Він дозволяє комплексно автоматизувати вирішення задачі обліку і управління ні підприємствах громадського харчування.

ПП автоматизує роботу одразу декількох віділів закладу громадського харчування, а саме: бухгалтера, бармена, кухаря. Створено на базі «1С-Предприятие». «іОбщепит» дозволяє швидко і правильно оформити замовлення на гостей, проконтролювати його видачу і швидко розрахувати клієнта. «іОбщепит» дозволяє не тільки вести облік по калькуляційних картах, але й розробляти їх. Дозволяє вводити окремі факти приготування страв і заготовки напівфабрикатів

Недоліками даного ПП є великі вимоги до апаратної частини ПК. Часто для коректної роботи використовують термінальні сервери. Тому використання даного ПП є досить дорогим, адже крім купівлі ліцензії iОбщепит, користувач повинен мати ПК з високими показниками апаратної частини або винаймати термінальний сервер. Крім цього у даному ПП функціонал автоматизування робочого місця кухаря урізаний, адже в основному наголос робиться на автоматизації робочого місця офіціанта.

Наступним рішенням є ПП РестАрт: АРМ Повара від компанії 1С-Rarus Ltd. Це мобільний додаток для планшетів на базі Android. Знаходиться у демо версії. У даному ПП при завантаженні є вбудована база страв з можливістю поповнення списку, рецептура, технологія приготування та вимоги зовнішнього вигляду страви. Щодо недоліків, функціонал ПП обмежується тільки приготуванням і подачею страв. Beta версія означає, що ПП знаходиться на стадії розробки, тому можливі баги при роботі [8].

На ринку існує небагато програмних продуктів для, які можуть ефективно оптимізувати роботу кухаря. Основним недоліком існуючих систем можна вважати обмежений функціонал. Тому доцільною є розробка програмного продукту направленого на автоматизацію робочого місця працівника кухні з можливістю регістрації приготування страв, прийняття поставок та виконання ревізій.

screenshots

1.2.2. Аналіз сучасних засобів створення програмного забезпечення

Для розробки програмного продукту направленого на автоматизацію робочого місця працівника кухні, перш за все потрібно обрати програмну платформу. Найбільш популярними є: Microsft .Net, Mono, Java.

Microsoft .NET  — програмна технологія, запропонована фірмою [Microsoft](https://uk.wikipedia.org/wiki/Microsoft) як платформа для створення як звичайних програм, так і [веб-застосунків](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). Багато в чому є продовженням ідей та принципів, покладених в технологію [Java](https://uk.wikipedia.org/wiki/Java). Одною з ідей .NET є сумісність служб, написаних різними [мовами](https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F). Хоча ця можливість рекламується Microsoft як перевага .NET, платформа Java має таку саму можливість.

Середовища розробки .NET-програм: [Visual Studio .NET](https://uk.wikipedia.org/wiki/Visual_Studio_.NET) (C++, C#, J#), [SharpDevelop](https://uk.wikipedia.org/w/index.php?title=SharpDevelop&action=edit&redlink=1), [Borland](https://uk.wikipedia.org/wiki/Borland) Developer Studio (Delphi, C#) тощо. Середовище [Eclipse](https://uk.wikipedia.org/wiki/Eclipse) має додаток для розробки .NET-програм. Застосовні програми також можна розроблювати в текстовому редакторі та використовувати консольний компілятор.

Програмна платформа Java – ряд програмних продуктів і специфікацій компанії Sun Microsystems, раніше незалежної компанії, а нині дочірньої компанії корпорації Oracle, які спільно надають систему для розробки прикладного програмного забезпечення та вбудовування її в будь-який крос-платформенне програмне забезпечення. Java використовується в самих різних комп'ютерних платформах від вбудованих пристроїв і мобільних телефонів в нижньому ціновому сегменті, до корпоративних серверів і суперкомп'ютерів у вищому ціновому сегменті. Хоча Java-аплети рідко використовуються в настільних комп'ютерах, проте вони в них іноді використовуються для поліпшення функціональності і підвищення безпеки при перегляді всесвітньої павутини.

Програмний код, написаний на Java, віртуальна машина Java перетворить в байт-код Java. Однак є компілятори байт-коду для інших мов програмування, таких як Ada, JavaScript, Python, і Ruby. Також є кілька нових мов програмування, розроблених для роботи з віртуальною машиною Java. Це такі мови як Scala, Clojure and Groovy. Синтаксис Java в основному запозичений з Сі і C ++, але об'єктно-орієнтовані можливості засновані на моделі, використовуваної в Smalltalk і Objective-C. У Java відсутні певні низькорівневі конструкції, такі як покажчики, також Java має дуже просту модель пам'яті, де кожен об'єкт розташований в купі і всі змінні об'єктного типу є посиланнями. Управління пам'яттю здійснюється за допомогою інтегрованої автоматичної збірки сміття, яку виконує JVM.

Mono — багато-платформове [вільне](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D0%BB%D1%8C%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B5_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F) [відкрите](https://uk.wikipedia.org/wiki/%D0%92%D1%96%D0%B4%D0%BA%D1%80%D0%B8%D1%82%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%B5_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F) втілення системи [.NET](https://uk.wikipedia.org/wiki/.NET), яке відповідає стандартам [ECMA](https://uk.wikipedia.org/wiki/ECMA), включаючи серед іншого і [компілятор](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%96%D0%BB%D1%8F%D1%82%D0%BE%D1%80) [C#](https://uk.wikipedia.org/wiki/C_Sharp), і [Common Language Runtime](https://uk.wikipedia.org/wiki/Common_Language_Runtime).

Mono включає [компілятор](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%96%D0%BB%D1%8F%D1%82%D0%BE%D1%80) мови [C#](https://uk.wikipedia.org/wiki/C_Sharp)  — mcs, середовище виконання [.NET](https://uk.wikipedia.org/wiki/.NET) — mono (із підтримкою [JIT](https://uk.wikipedia.org/wiki/JIT)) і mint (без підтримки [JIT](https://uk.wikipedia.org/wiki/JIT)), [зневаджувач](https://uk.wikipedia.org/wiki/%D0%97%D0%BD%D0%B5%D0%B2%D0%B0%D0%B4%D0%B6%D1%83%D0%B2%D0%B0%D1%87),і ряд [бібліотек](https://uk.wikipedia.org/wiki/%D0%91%D1%96%D0%B1%D0%BB%D1%96%D0%BE%D1%82%D0%B5%D0%BA%D0%B0_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)), включаючи реалізацію [ADO.NET](https://uk.wikipedia.org/wiki/ADO.NET) і [ASP.NET](https://uk.wikipedia.org/wiki/ASP.NET). У межах проекту також розробляються прив'язки для графічної системи [GTK+](https://uk.wikipedia.org/wiki/GTK%2B) на платформі .NET.

Середовище виконання mono може виконувати модулі, написані мовами [C#](https://uk.wikipedia.org/wiki/C_Sharp), [F#](https://uk.wikipedia.org/wiki/F_Sharp), [VisualBasic,.NET](https://uk.wikipedia.org/wiki/Visual_Basic_.NET), [Java](https://uk.wikipedia.org/wiki/Java), [Boo](https://uk.wikipedia.org/wiki/Boo), [Nemerle](https://uk.wikipedia.org/wiki/Nemerle), [Python](https://uk.wikipedia.org/wiki/Python),[Forth](https://uk.wikipedia.org/wiki/Forth), [JavaScript](https://uk.wikipedia.org/wiki/JavaScript), [PHP](https://uk.wikipedia.org/wiki/PHP) і [Object Pascal](https://uk.wikipedia.org/wiki/Object_Pascal) (за наявності компілятора у середовище .Net/Mono). Очікується також підтримка мов [C](https://uk.wikipedia.org/wiki/%D0%A1%D1%96_(%D0%BC%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)),[Ada 2005](https://uk.wikipedia.org/wiki/%D0%90%D0%B4%D0%B0) і [Eiffel](https://uk.wikipedia.org/wiki/%D0%95%D0%B9%D1%84%D0%B5%D0%BB%D1%8C_(%D0%BC%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)).

Таким чином, проаналізувавши описані вище програмні платформи, для розробки програмного продукту було обрано платформу Microsoft .Net.

1.3. Постановка задачі

Потрібно розробити, використовуючи технологію .Net, автоматизоване робоче місце кухаря, призначене для інформаційного супроводу приготування страв, прийому поставок, виконання ревізій.

Розроблювана система повина виконувати такі функції:

* 1. робота з базою страв (додавання, редагування, параметричний пошук, друк карток страв);
  2. робота з базою складових (додавання, редагування, параметричний пошук, друк списку складових);
  3. робота з базою інгрідієнтів (додавання, редагування, параметричний пошук, друк списку інгрідієнтів);
  4. робота з базою ревізій (проведення, перегляд списку, друк списку);
  5. робота з базою поставок (прийняття, перегляд списку, друк списку);
  6. робота з базою партій (прийняття, перегляд списку, друк списку);
  7. робота з базою постачальників (додавання, перегляд списку, друк списку);
  8. робота з базою руху інгрідієнтів (додавання, перегляд списку, друк списку);
  9. робота з базою кадрів (прийняття працівника, перегляд списку, друк списку, додавання посади, перегляд списку посад, параметричний пошук).

2. Проектування структури інформаційної системи

2.1. Аналіз та автоматизація обробки інформаційних потоків

Розроблюваний програмний продукт націлений на одну групу користувачів – кухарі. Робота з ним має проходити у декілька етапів, як показано на рисунку 2.1. На кожному етапі описується стадія заповнення бази, починаючи від занесення інформації про працівників та іншої допоміжної інформації – до контрольний функцій.

Занесення додаткової інформації

Реєстрація інгрідієнтів

Запис рецептів

Прийняття поставок

Створення карток страв

Контроль функцій

Рисунок 2.1 – Схема етапів роботи

На етапі внесення додаткової інформації в систему вносяться дані про працівників (П.І.Б., дата народження, дата прийняття на роботу, номер телефону, домашня адреса), статі, національності, кваліфікації, посади, вулиці, міста, країни, типи страв, складових, інгрідієнтів, вимірювання, кухні, види руху інгрідієнтів, напрям руху, види ревізій, постачальників.

На етапі реєстрації інгрідієнтів в систему вносяться дані про інгрідієнти: назва, тип, вимірювання.

На етапі рецептів в систему вносяться дані про страви (назва, тип, кухня, маса, ціна, кухар, рецепт) та складові (назва, страва, тип, коментар).

На етапі поставок в систему вносяться дані про поставки (дата/час, працівник, постачальник, сума), партії (порядковий номер, кількість, інгрідієнти, номер поставки, ціна за 1од).

На етапі приготування в систему вносять дані про приготування 1 страви (працівник, страва, дата/час початку, дата/час завершення).

На етапі контролю в систему вносять дані про ревізі (порядковий номер, дата/час початку, дата/час завершення, працівник, вид ревізії, коментар) та рух інгрідієнтів (дата/час, порядковий номер партії, вид руху, працівник, кількість, коментар).

Відповідно до розглянутих вище етапів роботи з системи, можна виділити наступні бізнес-процеси: «Приготування», «Страви», «Складові», «Інгрідієнти», «Ревізії», «Поставки», «Працівники».

Бізнес-процес дає можливість користувачеві отримати детальну інформацію про страви, які готують у відповідному закладі. Властивості: назва, тип, кухня, маса, ціна, працівник, рецепт – рисунок 2.2.

Страва

Рисунок 2.2 – Властивості бізнес-процесу «Страви»

Бізнес процес «Складові» дозволяє кухарю отримати інформацію про компоненти, з яких складається страва та їх приготування.Властивості: назва, страва, тип, опис – рисунок 2.3.

Складова

Рисунок 2.3 – Властивості бізнес-процесу «Складові»

Бізнес-процес «Інгрідієнти» дозволяє користувачеві отримати інформацію про інгрідієнти з яких складається страва, їх міру вимірювання та тип.Властивості: назва, тип, вимірювання – рисунок 2.4.

Інгрідієнти

Рисунок 2.4 – Властивості бізнес-процесу «Інгрідієнти»

Бізнес-процес «Ревізії» дає можливість користувачу перевіряти кількість використаних інгрідієнтів відповідно до кількості інгрідієнтів в наявності. Властивості: порядковий номер, дата/час початку, дата/час завершення, працівник, вид ревізії, коментар – рисунок 2.5.

Ревізії

Рисунок 2.5 – Властивості бізнес-процесу «Ревізії»

Бізнес-процес «Поставки» дозволяє користувачеві отримати інформацію про отримані поставки товарів. Властивості: порядковий номер, дата/час, працівник, постачальник, сума – рисунок 2.6.

Поставки

Рисунок 2.6 – Властивості бізнес-процесу «Поставки»

Бізнес-процес «Працівники» дозволяє користувачеві отримати повну інформацію по працівникам закладу. Властивості: П.І.Б., дата народження, дата прийняття, стать, національність, кваліфікація, номер телефону, посада, домашня адреса – рисунок 2.7.

Працівники

Рисунок 2.7 – Властивості бізнес-процесу «Працівники»

Бізнес процес «Приготування» дозволяє заносити користувачеві інформацію про приготовані страви. Властивості: страва, кухар, дата/час початку, дата/час завершення – рисунок 2.8.

Приготування

Рисунок 2.7 – Властивості бізнес-процесу «Працівники»

З огляду на проведений аналіз, функціонал АРМ Кухаря можна розбити на наступні групи функцій:

* + 1. Кухня – робота з базою приготування страв.

1. Рецептури – робота з базою страв, складових, інгрідієнтів.
2. Контроль – робота з базою ревізій, поставок, постачальників, партій, рухів.
3. Кадри – робота з базою працівників.

Група функцій «Кухня» включає наступні функції:

* 1. Створення нової картки страви.
  2. Перегляд списку приготованих страв.
  3. Вивід на друг списку приготованих страв.

Група функцій «Рецептури» включає наступні функції:

* + 1. Додавання нового інгрідієнту.
    2. Перегляд списку інгрідієнтів.
    3. Вивід на друк списку інгрідієнтів.
    4. Додавання нової страви.
    5. Перегляд списку страв.
    6. Вивід на друк списку страв.
    7. Додавання нової складової.
    8. Перегляд списку складових.
    9. Вивід на друк списку складових.

Група функцій «Контроль» включає наступні функції:

* + - 1. Проведення нової ревізії.
      2. Перегляд списку проведених ревізій.
      3. Вивід на друк списку проведених ревізій.
      4. Прийняття нової поставки.
      5. Перегляд списку поставок.
      6. Вивід на друк списку поставок.
      7. Додавання нового постачальника.
      8. Перегляд списку постачальників.
      9. Вивід на друк списку постачальників.
      10. Прийняття нової партії.
      11. Перегляд списку партій.
      12. Вивід на друк списку партій.
      13. Додавання нового руху інгрідієнту.
      14. Перегляд списку рухів інгрідієнтів.
      15. Вивід на друк списку інгрідієнтів.

Група функцій «Кадри» включає наступні функції:

* + - * 1. Додавання нового працівника.
        2. Перегляд списку працівників.
        3. Вивід на друк списку працівників.
        4. Додавання нової посади.
        5. Перегляд списку посад.
        6. Сформування списку працівників за посадами.

Зважаючи на наведені властивості сутностей та функції системи, можна розробити структуру відповідної бази даних, призначеної для збереження даних про страви, складові, інгрідієнти тощо.

2.2. Розробка структури інформаційної системи

Створення бази завжди починається з розробки структури її таблиць. Структура повина бути такою, щоб при роботі з базою потрібно було вводити в неї якнайменше даних. Якщо введення даних доводиться повторювати неодноразово, база робиться з декількох зв’язаних таблиць. Структура кожної таблиці розробляється окремо.

Для того, щоб зв’язки між таблицями працювали надійно і по запису однієї таблиці можна було однозначно знайти записи іншої, потрібно передбачити в таблиці унікальні поля. При створенні таблиць, необхідно визначити, які дані необхідно зберігати для конкретних таблиць.

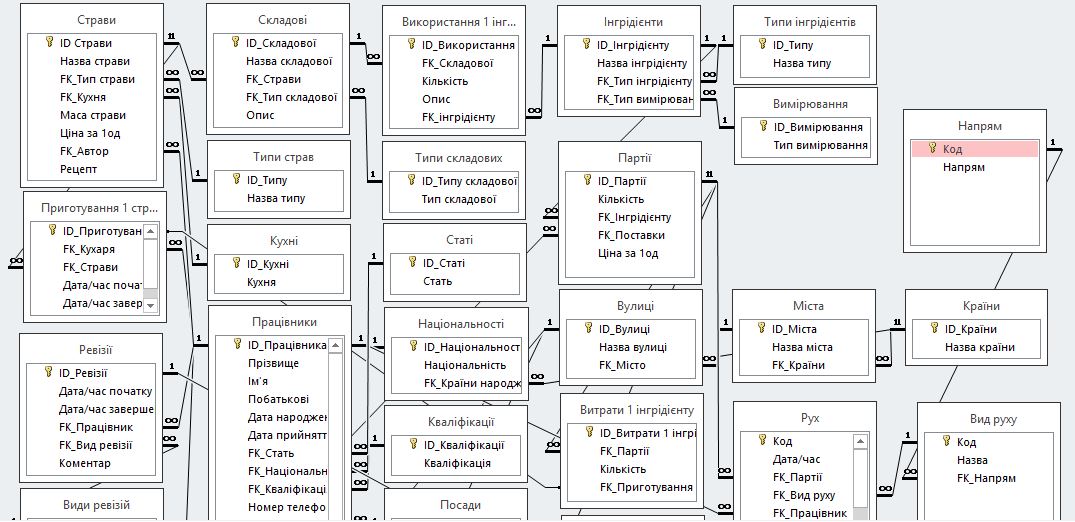


Рисунок 2.2 – Схема даних БД «АРМ Кухаря»

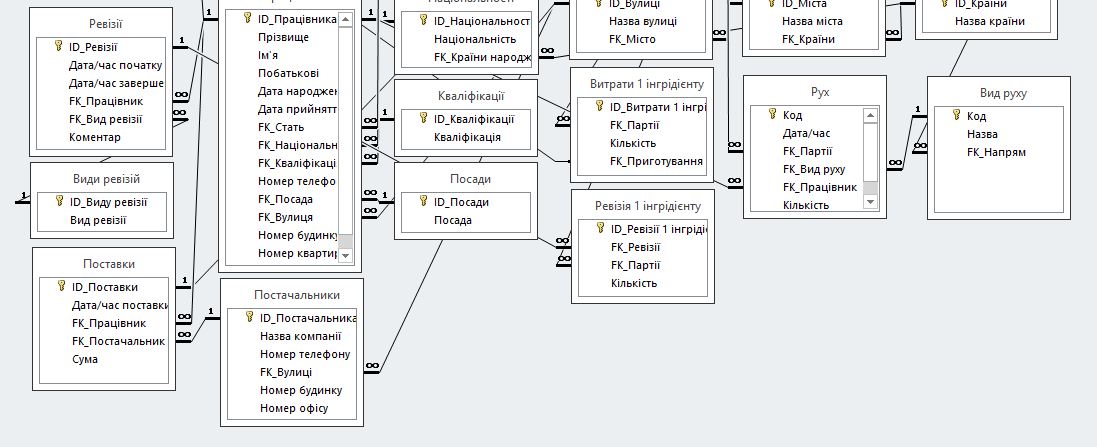


Рисунок 2.3 – Схема даних БД «АРМ Кухаря» (продовження)

Таблиця «Приготування 1 страви» призначена для зберігання інформації про приготовані кухарями страви. В цій таблиці мають бути наступні поля:

«ID\_Приготування» – первинний ключ таблиці.

«Дата/час початку» – дата/час початку приготування страви.

«Дата/час завершення» – дата/час завершення приготування страви.

«FK\_Страви» – вторинний ключ для визначення страви, яку приготували.

«FK\_Кухаря» – вторинний ключ для визначення кухаря, відповідавшого за приготування страви.

Таблиця «Страви» призначена для зберігання страв, приготування яких можливе згідно даних нашої БД. В цій таблиці мають бути наступні поля:

«ID Страви» – первинний ключ таблиці.

«Назва страви» – назва страви.

«FK\_Тип страви» – вторинний ключ для визначення типу страви.

«Маса страви» – маса страви.

«Ціна за 1од» – ціна страви за 1 одиницю.

«FK\_Автор» – вторинний ключ для визначення кухаря, який придумав страву.

«Рецепт» – рецепт для приготування страви.

Таблиця «Складові» призначена для зберігання складових, з яких в свою чергу складаються страви. В цій таблиці мають бути наступні поля:

«ID\_Складової» – первинний ключ таблиці.

«Назва складової» – назва складової.

«FK\_Страви» – вторинний ключ для визначення до якої страви відноситься складова.

«FK\_Тип складової» – вторинний ключ для визначення типу складової.

«Опис» – міні-рецепт приготування складової.

Таблиця «Інгрідієнти» призначена для зберігання інгрідієнтів, які згідно рецептів потрібні для приготування складової страви. В таблиці мають бути наступні поля:

«ID\_Інгрідієнту» – первинний ключ таблиці.

«Назва інгрідієнту» – назва інгрідієнту.

«FK\_Тип вимірювання» – вторинний ключ для визначення типу, в якому вимірюється інгрідієнт.

«FK\_Тип інгрідієнту» – вторинний ключ для визначення типу, до якого належить інгрідієнт.

Таблиця «Ревізії» призначена для зберігання даних, про проведення ревізії. В таблиці мають бути наступні поля:

«ID\_Ревізії» – первинний ключ таблиці.

«Дата/час початку» – дата/час початку проведення ревізії.

«Дата/час завершення» - дата/час завершення проведення ревізії.

«FK\_Працівник» – вторинний ключ для визначення працівника, який провів ревізію.

«FK\_Тип ревізії» – вторинний ключ для визначення типу проведеної ревізії.

«Коментар» – короткий опис проведеної ревізії.

Таблиця «Поставки» призначена для зберігання даних про прийняті поставки товарів. В таблиці мають бути наступні поля:

1. «ID\_Поставки» – первинний ключ таблиці.
2. «Дата/час поставки» – дата/час прийняття поставки.
3. «FK\_Працівник» – вторинний ключ для визначення працівника, прийнявшого поставку.
4. «FK\_Постачальник» – вторинний ключ для визначення постачальника, який відвантажив поставку.
5. «Сума» – сума поставки.

Таблиця «Рух» призначена для зберігання даних про рух інгрідієнтів всередині кухні. В таблиці мають бути наступні поля:

1. «ID\_Руху» – первинний ключ таблиці.
2. «Дата/час» – дата/час проведення руху партії.
3. «FK\_Партії» – вторинний ключ для визначення партії, рух якої проводиться.
4. «FK\_Вид руху» – вторинний ключ для визначення напряму руху партії.
5. «FK\_Працівник» – вторинний ключ для визначення працівника, провівшого рух партії.
6. «Кількість» – кількість інгрідієнтів партії, які підверглися руху.
7. «Коментар» – короткий опис проведеного руху.

Таблиця «Працівники» призначена для зберігання даних про працівників. В таблиці мають бути наступні поля:

1. «ID\_Працівника» – первинний ключ таблиці.
2. «Прізвище» – прізвище працівника.
3. «Ім’я» – ім’я працівника.
4. «По батькові» – по батькові працівника.
5. «Дата народження» – дата народження працівника.
6. Дата прийняття» – дата прийняття працівника на кухню.
7. «FK\_Стать» – вторинний ключ для визначення статі працівника.
8. «FK\_Національність» – вторинний ключ для визначення національності працівника.
9. «FK\_Кваліфікація» – вторинний ключ для визначення освіти та наукогового ступеня працівника.
10. «Номер телефона» – номер телефона працівника.
11. «FK\_Посада» – вторинний ключ для визначення посади працівника.
12. «FK\_Вулиця» – вторинний ключ для визначення домашньої адреси працівника.
13. «Номер будинку» – домашня адреса працівника.
14. «Номер квартири» – домашня адреса працівника.
15. «Попередній стаж» – сума кількості років роботи на кухні на попередніх місцях роботи.

Таким чином, в БД зберігається вся інформація, яка потрібна для автоматизації робочого місця кухаря.

Основні функції бізнес-процесій за допомогою цих таблиць

Для реал функції юізнес-процесу виконуємо…

2.3. Вибір засобів розробки інформаційної системи

2.3.1. Вибір мови програмування

Для розробки застосунку, призначеного для занесення до системи даних про заплановані заняття, можна використати мови програмування С# або С++.

***C++*** – мова програмування високого рівня з підтримкою декількох парадигм програмування: об’єктно-орієнтованої, узагальненої та процедурної. Базується на мові С.

У 1990-х роках С++ стала однією з найуживаніших мов програмування загального призначення. Мову використовують для системного програмування, розробки програмного забезпечення, написання драйверів, потужних серверних та клієнтських програм, а також для розробки розважальних програм таких як відеоігри. С++ суттєво вплинула на інші, популярні сьогодні, мови програмування: [С#](http://uk.wikipedia.org/wiki/C_Sharp) та [Java](http://uk.wikipedia.org/wiki/Java) [11].

***C#*** – [об'єктно-орієнтована](http://uk.wikipedia.org/wiki/%D0%9E%D0%B1%E2%80%99%D1%94%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D1%96%D1%94%D0%BD%D1%82%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) [мова програмування](http://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) з безпечною [системою типізації](http://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%82%D0%B8%D0%BF%D1%96%D0%B7%D0%B0%D1%86%D1%96%D1%97) для платформи[.NET](http://uk.wikipedia.org/wiki/.NET).

[Синтаксис](http://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BD%D1%82%D0%B0%D0%BA%D1%81%D0%B8%D1%81) C# близький до [С++](http://uk.wikipedia.org/wiki/%D0%A1%2B%2B) і [Java](http://uk.wikipedia.org/wiki/Java). [Мова](http://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B2%D0%B0) має строгу статичну типізацію, підтримує [поліморфізм](http://uk.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D1%96%D0%BC%D0%BE%D1%80%D1%84%D1%96%D0%B7%D0%BC_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)), перевантаження операторів, вказівники на функції-члени класів, атрибути, події, властивості, [винятки](http://uk.wikipedia.org/wiki/%D0%9E%D0%B1%D1%80%D0%BE%D0%B1%D0%BA%D0%B0_%D0%B2%D0%B8%D0%BD%D1%8F%D1%82%D0%BA%D1%96%D0%B2), коментарі у форматі [XML](http://uk.wikipedia.org/wiki/XML). Перейнявши багато що від своїх попередників – мов [С++](http://uk.wikipedia.org/wiki/%D0%A1%2B%2B), [Delphi](http://uk.wikipedia.org/wiki/Delphi_(%D0%BC%D0%BE%D0%B2%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)), [Модула](http://uk.wikipedia.org/w/index.php?title=%D0%9C%D0%BE%D0%B4%D1%83%D0%BB%D0%B0&action=edit&redlink=1) і [Smalltalk](http://uk.wikipedia.org/wiki/Smalltalk) С#, спираючись на практику їхнього [використання](http://uk.wikipedia.org/wiki/%D0%92%D0%B8%D0%BA%D0%BE%D1%80%D0%B8%D1%81%D1%82%D0%B0%D0%BD%D0%BD%D1%8F), виключає деякі моделі, що зарекомендували себе як проблематичні при розробці програмних систем, наприклад [множинне спадкування](http://uk.wikipedia.org/wiki/%D0%9C%D0%BD%D0%BE%D0%B6%D0%B8%D0%BD%D0%BD%D0%B5_%D1%81%D0%BF%D0%B0%D0%B4%D0%BA%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F) класів (на відміну від C++) [12].

За результатами проведеного аналізу, для розробки застосунку, призначеного для занесення до системи даних про заплановані заняття, було обрано мову програмування С#.

Тільки вибір

2.3.2. Вибір системи керування базами даних

В наш час є ряд СКБД, використовуючи які можна реалізувати автоматизовану інформаційну систему. Для вибору найбільш прийнятного варіанту було розглянуто деякі з них.

*Microsoft SQL Server* – комерційна [система керування базами даних](http://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85), що розповсюджується корпорацією Microsoft. Мова, що використовується для запитів – [Transact-SQL](http://uk.wikipedia.org/w/index.php?title=Transact-SQL&action=edit&redlink=1), створена спільно Microsoft та [Sybase](http://uk.wikipedia.org/wiki/Sybase). Transact-SQL є реалізацією стандарту [ANSI](http://uk.wikipedia.org/wiki/ANSI)/[ISO](http://uk.wikipedia.org/wiki/ISO) щодо структурованої мови запитів ([SQL](http://uk.wikipedia.org/wiki/SQL)) із розширеннями. Використовується як для невеликих і середніх за розміром баз даних, так і для великих баз даних масштабу підприємства. Багато років вдало конкурує з іншими системами керування базами даних [13].

*«Microsoft Access»* (повна назва Microsoft Office Access) – [система управління базами даних](http://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D1%96%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) від компанії [Майкрософт](http://uk.wikipedia.org/wiki/%D0%9C%D0%B0%D0%B9%D0%BA%D1%80%D0%BE%D1%81%D0%BE%D1%84%D1%82),[програма](http://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%B0), що входить до складу [пакету офісних програм](http://uk.wikipedia.org/wiki/%D0%9E%D1%84%D1%96%D1%81%D0%BD%D0%B8%D0%B9_%D0%BF%D0%B0%D0%BA%D0%B5%D1%82) [Microsoft Office](http://uk.wikipedia.org/wiki/Microsoft_Office). Має широкий спектр функцій, включаючи зв'язані запити, сортування по різних полях, зв'язок із зовнішніми таблицями і базами даних. Завдяки вбудованій мові VBA, в самому Access можна писати підпрограми, що працюють з базами даних [14].

СКБД Access входить до складу широко розповсюдженого сімейства офісних додатків Microsoft Office. Microsoft Access на сьогоднішній день є одним з самих розповсюджених настільних додатків для роботи с базами даних. Це пов’язано з тим, що Access володіє дуже широким спектром засобів для вводу, аналізу і представлення даних. Ці засоби є не тільки простими і зручними, але й високопродуктивними, що забезпечують високу швидкість розробки додатків.

Удосконалюючись від версії до версії Access став інструментом, який може задовольнити потреби найрізноманітніших категорій користувачів: від початківця, якому подобається дружній інтерфейс системи, який дозволяє йому справитись із задачами, до професійного розробника, який має весь необхідний інструментарій для побудови унікального рішення для конкретного підприємства середнього бізнесу.

Microsoft Access дозволяє керувати всіма відомостями в межах одного файлу бази даних. У рамках цього файлу дані можна розділити на окремі контейнери, які називаються таблицями; переглядати, додавати й поновлювати дані в таблицях за допомогою електронних форм; знаходити і витягати тільки потрібні дані за допомогою запитів; а також аналізувати чи друкувати дані в заданому макеті за допомогою звітів. Створення сторінок доступу до даних дозволяє користувачам переглядати, поновлювати чи аналізувати дані з бази даних через Інтернет.

MS Access має потужний, зручний і гнучкий засіб візуального проектування об’єктів за допомогою Майстрів, і це дає змогу при мінімумі попередньої підготовки швидко створити повноцінну базу даних – на рівні таблиць, форм, запитів-вибірок і звітів.

За допомогою програми Access можна:

1. додавати нові дані до бази даних, наприклад новий предмет до інвентарного списку;
2. редагувати наявні дані бази даних, наприклад змінювати поточне розташування предмета;
3. видаляти відомості, якщо, наприклад, предмет було продано або списано;
4. впорядковувати та переглядати дані різними способами;
5. спільно користуватися даними з іншими користувачами за допомогою звітів, повідомлень електронної пошти, інтрамережі або Інтернету.

Частини бази даних Access наведено нижче.

Таблиці бази даних схожі на електронну таблицю, в якій дані зберігаються в рядках і стовпцях. В результаті зазвичай досить легко імпортувати електронну таблицю до таблиці бази даних. Головна відмінність між збереженням даних в електронній таблиці та базі даних — це спосіб упорядкування даних.

Форми іноді називаються «екранами вводу даних». Це інтерфейси, які використовуються під час роботи з даними, тому вони часто містять кнопки для виконання різних команд. Можна створити базу даних без використання форм, просто редагуючи дані в таблицях даних. Проте більшість користувачів баз даних використовують форми для перегляду, введення та редагування даних у таблицях.

Звіти – використовуються для зведення та представлення даних у таблицях. Звіт зазвичай відповідає на певне питання, наприклад «Яку суму було отримано від кожного клієнта цього року?» або «У яких містах розташовані наші клієнти?». кожний звіт можна відформатувати таким чином, щоб він представляв дані найбільш зрозумілим способом.

Запити –  це справжні робочі коники бази даних, які можуть виконувати багато різних функцій. Їх найпоширеніша функція — отримання певних даних із таблиць. Дані, які потрібно переглянути, зазвичай розташовані в кількох таблицях, і запити дають змогу переглянути їх в одній таблиці даних. Також, оскільки зазвичай не потрібно бачити всі записи одночасно, запити дозволяють додавати критерії для «фільтрування» даних, щоб переглядати лише потрібні записи. Запити часто виконують роль джерела записів для форм і звітів.

Макроси в програмі Access можна вважати спрощеною мовою програмування, яку можна використовувати для додавання функціональності до бази даних. Наприклад, можна вкласти макрос до кнопки форми, щоб запускати макрос у разі натискання цієї кнопки. Макроси містять дії, які виконують завдання, наприклад відкривають звіт, виконують запит або закривають базу даних. Більшість операцій із базою даних, які виконуються вручну, можна зробити автоматичними за допомогою макросів, тому вони можуть бути корисним засобом економії часу.

Модулі, як і макроси, – це об’єкти, які можна використовувати для додавання функціональності до бази даних. Проте, якщо макроси Access створюються за допомогою вибору зі списку дій макросу, модулі пишуться мовою програмування [Visual Basic для застосунків (VBA)](javascript:AppendPopup(this,'824217560_1')). Модуль — це збірка декларацій, інструкцій і процедур, які зберігаються разом. Модуль може бути модулем класу або стандартним модулем. Модулі класу додаються до форм або звітів і зазвичай містять процедури, характерні для форми чи звіту, до яких вони додаються. Стандартні модулі містять загальні процедури, не пов’язані з жодним іншим об’єктом [15].

*СКБД Oracle* – це одна з найпоширеніших на сьогоднішній день систем управління баз даних. В СКБД Oracle виконуються всі сучасні вимоги, такі як: багатокористувацький доступ до системи клієнт-сервер, багатоступеневий захист від несанкціонованого доступу, реалізація принципу незалежності даних та ін. В даний час СКБД Oracle працює практично на всіх ОС. Головне, що характеризує продукти Oracle протягом багатьох років - це надійність, безпека, висока продуктивність, зручність у роботі. Для СКБД Oracle, що стала на сьогоднішній день практично обов'язковою частиною будь-якої серйозної інформаційної системи, це особливо важливо. Але не лише ці характеристики дозволяють продуктам Oracle утримувати лідерство на ринку СКБД. Інформаційні технології стрімко розвиваються та вимагають від сучасних СКБД розширення функціональності зі зберігання й обробки даних. Корпорація Oracle, рухаючись в ногу з часом, ламає погляди на СКБД, що склалися, наділяючи її все новими та новими можливостями.

Сучасна СКБД Oracle – це потужний програмний комплекс, який дозволяє створювати програми будь-якого ступеня складності. Ядром цього програмного комплексу є база даних, яка зберігає інформацію, причому кількість інформації за рахунок засобів масштабування, що надаються, практично безмежне. З цією інформацією високоефективно може одночасно працювати практично будь-яка кількість користувачів (за наявності достатніх апаратних ресурсів), причому продуктивність системи не знижується.

В останніх версіях СКБД Oracle механізми масштабування дозволяють безмежно збільшувати потужність і швидкість роботи сервера Oracle та програм за допомогою елементарного додавання нових вузлів кластера. Це не вимагає зупинки працюючих застосування чи переписування старих програм, розроблених для звичайної одномашинної архітектури. Крім того, вихід з ладу окремих вузлів кластера також не призводить до зупинки програми.  
 Багатоплатформність – ще одна складова успіху СКБД Oracle, так як вона поставляється практично для всіх існуючих на сьогодні операційних систем. При роботі на будь-якій операційній системі з продуктами Oracle не виникає жодних проблем: СКБД Oracle однаково добре працює на будь-якій платформі. Тому компаніям, які розпочинають роботу з продуктами Oracle, не доводиться міняти вже встановлене мережеве оточення. При роботі з СКБД Oracle існує лише невелика кількість відмінностей, зумовлених особливостями тієї чи іншої операційної системи. Але в цілому - це завжди найбезпечніша, надійна та зручна СКБД Oracle.

СКБД Oracle поставляється в чотирьох варіантах: Oracle Database Enterprise Edition, Oracle Database Standard Edition, Oracle Database Personal Edition і в зовсім полегшеному мобільному варіанті, призначеному, в першу чергу, для ноутбуків. При цьому всі варіанти сервера Oracle функціонально ідентичні, за винятком деяких опцій [16].

Порівнявши встановлені недоліки й переваги розглянутих СКБД, було обрано Microsoft SQL Server для подальшої розробки бази даних автоматизованої інформаційної системи ведення розкладу занять у ВНЗ.

Таким чином, для розробки було обрано комбінацію – мову С# для розробки застосунку та систему керування базами даних SQL Server для реалізації бази даних системи.

Тільки вибір

3. Програмна реалізація

3.1. Структура і функціональне призначення модулів системи, їх взаємозв’язок

В рамках створення автоматизованого робочого місця, необхідно розробити засобами СКБД відповідно до предметної області компоненти – таблиці та запити.

Всі функції, що мають виконуватись системою, реалізуються за допомогою вказаних компонентів. Так, компоненти для реалізації групи функцій по роботі з картками страв подано у таблиці 3.1.

Таблиця 3.1 – Група функцій по роботі з картками страв

|  |  |  |
| --- | --- | --- |
| **№ з.п.** | **Функція** | **Тип об’єкта, що викликається** |
| 1. | Створення нової картки страви | форма + запит |
| 2. | Вивід на друк списку створенних карток | запит + звіт |
| 3. | Перегляд списку створенних карток страв | форма + запит |

Група функцій по роботі рецептурами складається з функціональних блоків. Функціональний блок по роботі з інгрідієнтами подано у таблиці 3.2.

Таблиця 3.2 – Функціональний блок по роботі з інгрідієнтами

|  |  |  |
| --- | --- | --- |
| **№ з.п.** | **Функція** | **Тип об’єкта, що викликається** |
| 1. | Додавання нового інгрідієнту | форма + запит |
| 2. | Вивід на друк списку усіх інгрідієнтів | запит + звіт |
| 3. | Перегляд всього списку інгрідієнтів | форма + запит |

Функціональний блок по роботі з складовими подано у таблиці 3.3.

Таблиця 3.3 – Функціональний блок по роботі з складовими

|  |  |  |
| --- | --- | --- |
| **№ з.п.** | **Функція** | **Тип об’єкта, що викликається** |
| 1. | Додавання нової складової | форма + запит |
| 2. | Вивід на друк списку усіх складових | запит + звіт |
| 3. | Перегляд всього списку складових | форма + запит |

Функціональний блок по роботі зі стравами подано у таблиці 3.4.

Таблиця 3.4 – Функціональний блок по роботі з стравами

|  |  |  |
| --- | --- | --- |
| **№ з.п.** | **Функція** | **Тип об’єкта, що викликається** |
| 1. | Додавання нової страви | форма + запит |
| 2. | Вивід на друк списку усіх страв | запит + звіт |
| 3. | Перегляд всього списку страв | форма + запит |

Група функцій «Контроль» складається з функціональних блоків. Функціональний блок по роботі з ревізіями подано у таблиці 3.5.

Таблиця 3.5 – Функціональний блок по роботі з ревізіями

|  |  |  |
| --- | --- | --- |
| **№ з.п.** | **Функція** | **Тип об’єкта, що викликається** |
| 1. | Проведення ревізії | форма + запит |
| 2. | Вивід на друк списку проведених ревізій | запит + звіт |
| 3. | Перегляд всього списку проведених ревізій | форма + запит |

Функціональний блок по роботі з поставками подано у таблиці 3.6.

Таблиця 3.6 – Функціональний блок по роботі з поставками

|  |  |  |
| --- | --- | --- |
| **№ з.п.** | **Функція** | **Тип об’єкта, що викликається** |
| 1. | Прийняття поставки | форма + запит |
| 2. | Вивід на друк списку усіх поставкок | запит + звіт |
| 3. | Перегляд всього списку поставок | форма + запит |

Функціональний блок по роботі з постачальниками подано у таблиці 3.7.

Таблиця 3.7– Функціональний блок по роботі з постачальниками

|  |  |  |
| --- | --- | --- |
| **№ з.п.** | **Функція** | **Тип об’єкта, що викликається** |
| 1. | Знаходження нового постачальника | форма + запит |
| 2. | Вивід на друк списку усіх постачальників | запит + звіт |
| 3. | Перегляд всього списку постачальників | форма + запит |

Функціональний блок по роботі з ревізіями подано у таблиці 3.8.

Таблиця 3.8– Функціональний блок по роботі з ревізіями

|  |  |  |
| --- | --- | --- |
| **№ з.п.** | **Функція** | **Тип об’єкта, що викликається** |
| 1. | Проведення ревізії | форма + запит |
| 2. | Вивід на друк списку усіх проведених ревізій | запит + звіт |
| 3. | Перегляд всього списку проведених ревізій | форма + запит |

Компоненти для реалізації групи функцій по роботі з кадрами подано у таблиці 3.9.

Таблиця 3.9 – Група функцій по роботі з кадрами

|  |  |  |
| --- | --- | --- |
| **№ з.п.** | **Функція** | **Тип об’єкта, що викликається** |
| 1. | Додавання нового працівника | форма + запит |
| 2. | Перегляд списку працівників | форма + запит |
| 3. | Вивід на друк особової справи працівника | звіт + запит |
| 4. | Додавання нової посади | форма + запит |
| 5. | Перегляд списку посад | форма + запит |
| 6. | Сформувати список працівників за посадою | запит + звіт |

Основними функціональними формами, що викликаються з головної форми, є форми роботи картками страв, стравами, складовими, інгрідієнтами, поставками, постачальниками, ревізіями, працівниками.

Таким чином, для функціонування автоматизованого робочого місця треба розробити компоненти Microsоft SQL Server у визначеному обсязі.

Для розробки застосунку розроблено діаграму класів – рисунок 3.1.

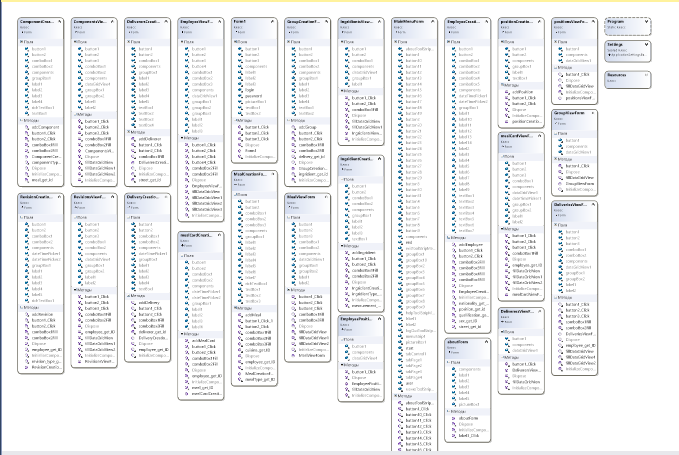


Рисунок 3.1 – Діаграма класів застосунку

ComponentCreationForm – клас, який наслідує клас Form та служить для додавання у БД компонентів за допомогою графічного інтерфейсу.

ComponentViewForm – клас, який наслідує клас Form та служить для відображення компонентів з БД на форму за допомогою DataGridView.

DeliverersCreationForm – клас, який наслідує клас Form та служить для додавання у БД постачальників за допомогою графічного інтерфейсу.

DeliverersViewForm – клас, який наслідує клас Form та служить для відображення списку постачальників з БД на форму за допомогою DataGridView.

DeliveryCreationForm – клас, який наслідує клас Form та служить для додавання у БД поставок за допомогою графічного інтерфейсу.

DeliveriesViewForm – клас, який наслідує клас Form та служить для відображення списку поставок з БД на форму за допомогою DataGridView.

EmployeeCreationForm – клас, який наслідує клас Form та служить для додавання у БД працівників за допомогою графічного інтерфейсу.

EmployeeViewForm – клас, який наслідує клас Form та служить для відображення списку працівників з БД на форму за допомогою DataGridView.

EmployeePositionGenerateList – клас, який наслідує клас Form та служить для відображення списку працівників за посадами з БД на форму за допомогою DataGridView

Form1 – клас, який наслідує клас Form та служить для авторизації користувача у системі.

GroupCreationForm – клас, який наслідує клас Form та служить для додавання у БД партій за допомогою графічного інтерфейсу.

GroupViewForm – клас, який наслідує клас Form та служить для відображення списку партій з БД на форму за допомогою DataGridView.

IngridientCreationForm – клас, який наслідує клас Form та служить для додавання у БД інгрідієнтів за допомогою графічного інтерфейсу.

IngridientViewForm – клас, який наслідує клас Form та служить для відображення списку інгрідієнтів з БД на форму за допомогою DataGridView.

MainMenuForm – клас, який наслідує клас Form та служить для відображення основних елементів інтерфейсу системи за допомого groupBox, button, label, tabControl.

MealCardCreationForm – клас, який наслідує клас Form та служить для додавання у БД карток страв за допомогою графічного інтерфейсу.

MealCardViewForm – клас, який наслідує клас Form та служить для відображення списку карток страв з БД на форму за допомогою DataGridView.

MealCreationForm – клас, який наслідує клас Form та служить для додавання у БД страв за допомогою графічного інтерфейсу.

MealViewForm – клас, який наслідує клас Form та служить для відображення списку страв з БД на форму за допомогою DataGridView.

PositionCreationForm – клас, який наслідує клас Form та служить для додавання у БД посад працівників за допомогою графічного інтерфейсу.

PositionViewForm – клас, який наслідує клас Form та служить для відображення списку посад працівників з БД на форму за допомогою DataGridView.

RevisionCreationForm – клас, який наслідує клас Form та служить для додавання у БД ревізій за допомогою графічного інтерфейсу.

RevisionViewForm – клас, який наслідує клас Form та служить для відображення списку ревізій з БД на форму за допомогою DataGridView.

3.2. Розробка програмних модулів

3.2.1. Створення таблиць для збереження даних

Для нормально функціонування застосунку було створенно БД, яка складається з 30 таблиць. Головними з них є: «Приготування 1 страви», «Страви», «Складові», «Інгрідієнти», «Ревізії», «Поставки», «Рух», «Працівники». Нижче описано поля таблиць, типи даних та їх призначення.

Таблиця «Приготування 1 страви» призначена для зберігання інформації про приготовані кухарями страви. В цій таблиці мають бути наступні   
поля (рис. 3.1, 3.2):

«ID\_Приготування» – первинний ключ таблиці.

«Дата/час початку» – дата/час початку приготування страви.

«Дата/час завершення» – дата/час завершення приготування страви.

«FK\_Страви» – вторинний ключ для визначення страви, яку приготували.

«FK\_Кухаря» – вторинний ключ для визначення кухаря, відповідавшого за приготування страви.

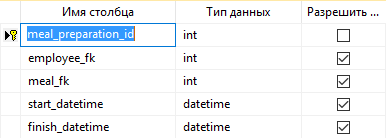


Рисунок 3.1 – Типи даних таблиці «Приготування 1 страви»

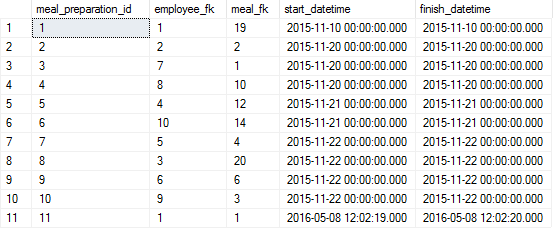


Рисунок 3.2 – Таблиця «Приготування 1 страви»

Таблиця «Страви» призначена для зберігання страв, приготування яких можливе згідно даних нашої БД. В цій таблиці мають бути наступні поля (рис. 3.3, 3.4):

«ID Страви» – первинний ключ таблиці.

«Назва страви» – назва страви.

«FK\_Тип страви» – вторинний ключ для визначення типу страви.

«Маса страви» – маса страви.

«Ціна за 1од» – ціна страви за 1 одиницю.

«FK\_Автор» – вторинний ключ для визначення кухаря, який придумав страву.

«Рецепт» – рецепт для приготування страви.

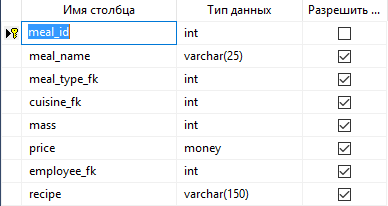


Рисунок 3.3 – Типи даних таблиці «Страви»

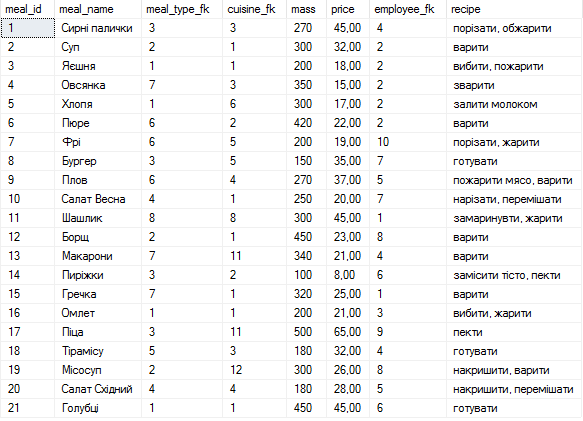


Рисунок 3.4 – Таблиця «Страви»

Таблиця «Складові» призначена для зберігання складових, з яких в свою чергу складаються страви. В цій таблиці мають бути наступні поля(рис. 3.5, 3.6):

«ID\_Складової» – первинний ключ таблиці.

«Назва складової» – назва складової.

«FK\_Страви» – вторинний ключ для визначення до якої страви відноситься складова.

«FK\_Тип складової» – вторинний ключ для визначення типу складової.

«Опис» – міні-рецепт приготування складової.

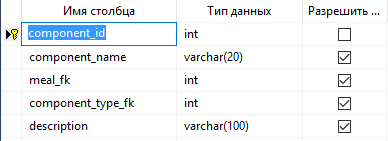


Рисунок 3.5 – Типи даних таблиці «Складові»



Рисунок 3.6 – Таблиця «Складові»

Таблиця «Інгрідієнти» призначена для зберігання інгрідієнтів, які згідно рецептів потрібні для приготування складової страви. В таблиці мають бути наступні поля(рис. 3.7, 3.8):

«ID\_Інгрідієнту» – первинний ключ таблиці.

«Назва інгрідієнту» – назва інгрідієнту.

«FK\_Тип вимірювання» – вторинний ключ для визначення типу, в якому вимірюється інгрідієнт.

«FK\_Тип інгрідієнту» – вторинний ключ для визначення типу, до якого належить інгрідієнт.

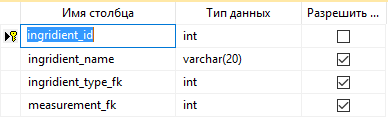


Рисунок 3.7 – Типи даних таблиці «Інгрідієнти»



Рисунок 3.8 – Таблиця «Інгрідієнти»

Таблиця «Ревізії» призначена для зберігання даних, про проведення ревізії. В таблиці мають бути наступні поля(рис. 3.9, 3.10):

«ID\_Ревізії» – первинний ключ таблиці.

«Дата/час початку» – дата/час початку проведення ревізії.

«Дата/час завершення» - дата/час завершення проведення ревізії.

«FK\_Працівник» – вторинний ключ для визначення працівника, який провів ревізію.

«FK\_Тип ревізії» – вторинний ключ для визначення типу проведеної ревізії.

«Коментар» – короткий опис проведеної ревізії.

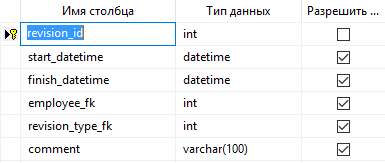


Рисунок 3.9 – Типи даних таблиці «Ревізії»

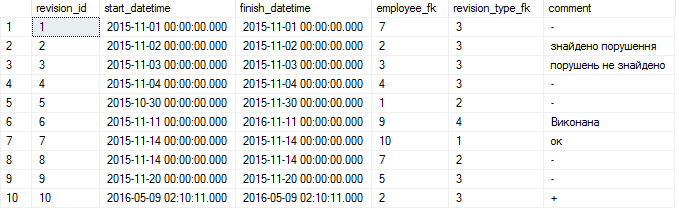


Рисунок 3.10 – Таблиця «Ревізії»

Таблиця «Поставки» призначена для зберігання даних про прийняті поставки товарів. В таблиці мають бути наступні поля(рис. 3.11, 3.12):

1. «ID\_Поставки» – первинний ключ таблиці.
2. «Дата/час поставки» – дата/час прийняття поставки.
3. «FK\_Працівник» – вторинний ключ для визначення працівника, прийнявшого поставку.
4. «FK\_Постачальник» – вторинний ключ для визначення постачальника, який відвантажив поставку.
5. «Сума» – сума поставки.

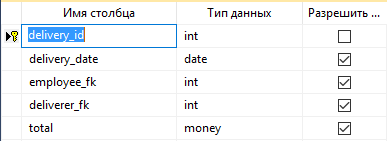


Рисунок 3.11 – Типи даних таблиці «Поставки»

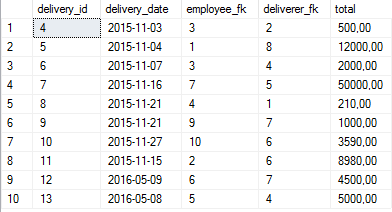


Рисунок 3.12 – Таблиця «Поставки»

Таблиця «Рух» призначена для зберігання даних про рух інгрідієнтів всередині кухні. В таблиці мають бути наступні поля(рис. 3.13, 3.14):

1. «ID\_Руху» – первинний ключ таблиці.
2. «Дата/час» – дата/час проведення руху партії.
3. «FK\_Партії» – вторинний ключ для визначення партії, рух якої проводиться.
4. «FK\_Вид руху» – вторинний ключ для визначення напряму руху партії.
5. «FK\_Працівник» – вторинний ключ для визначення працівника, провівшого рух партії.
6. «Кількість» – кількість інгрідієнтів партії, які підверглися руху.
7. «Коментар» – короткий опис проведеного руху.

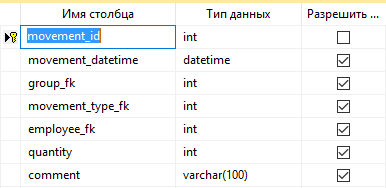


Рисунок 3.13 – Типи даних таблиці «Рух»

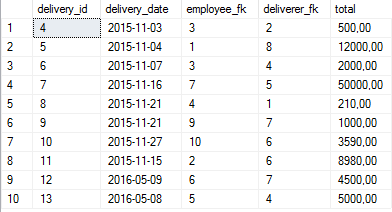


Рисунок 3.14 – Таблиця «Рух»

Таблиця «Працівники» призначена для зберігання даних про працівників. В таблиці мають бути наступні поля(рис. 3.15, 3.16):

1. «ID\_Працівника» – первинний ключ таблиці.
2. «Прізвище» – прізвище працівника.
3. «Ім’я» – ім’я працівника.
4. «По батькові» – по батькові працівника.
5. «Дата народження» – дата народження працівника.
6. Дата прийняття» – дата прийняття працівника на кухню.
7. «FK\_Стать» – вторинний ключ для визначення статі працівника.
8. «FK\_Національність» – вторинний ключ для визначення національності працівника.
9. «FK\_Кваліфікація» – вторинний ключ для визначення освіти та наукогового ступеня працівника.
10. «Номер телефона» – номер телефона працівника.
11. «FK\_Посада» – вторинний ключ для визначення посади працівника.
12. «FK\_Вулиця» – вторинний ключ для визначення домашньої адреси працівника.
13. «Номер будинку» – домашня адреса працівника.
14. «Номер квартири» – домашня адреса працівника.
15. «Попередній стаж» – сума кількості років роботи на кухні на попередніх місцях роботи.

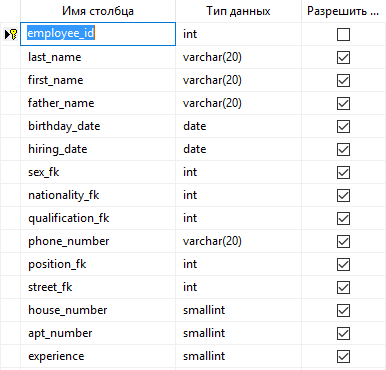


Рисунок 3.15 – Типи даних таблиці «Працівники»

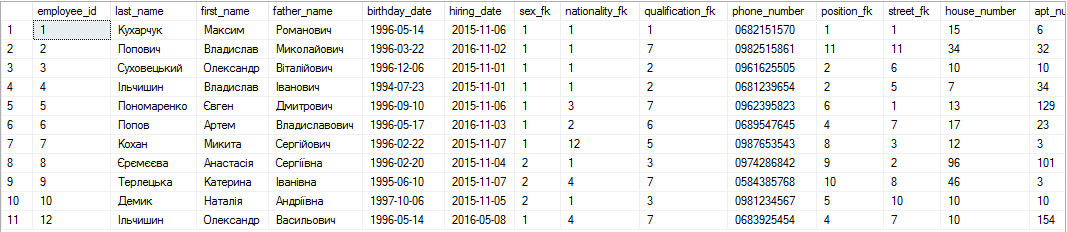


Рисунок 3.16 – Таблиця «Працівники»

Таким чином, для функціонування автоматизованої інформаційної системи треба розробити компоненти Microsоft SQL Server у визначеному обсязі.

3.2.2. Проектування запитів для рішення задач

Запит (query) – набір інструкцій, який можна використовувати для роботи з даними. Питання, сформоване стосовно бази даних, і є запит. Застосовуються два типи запитів: за зразком і структурована мова запитів (SQL).

SQL – структурна мова програмування, яка застосовується для створення, модифікації та управління даними в довільній реляційній базі даних, керованою відповідною системою управління базами даних (СКБД).

SQL використовується для опису наборів даних, які допомагають відповідати на запитання. Працюючи з SQL, потрібно зважати на правильність синтаксису. Синтаксис – набір правил, за якими визначається правильне поєднання елементів мови. Синтаксис SQL ґрунтується на синтаксисі англійської мови та застосовує багато спільних елементів із синтаксисом [Visual Basic для застосунків (VBA)](javascript:AppendPopup(this,'378130234_1')).

Для розробки БД «АРМ Кухаря» було використано наступні запити:

* Запити на вибір даних з БД – рисунок 3.17;

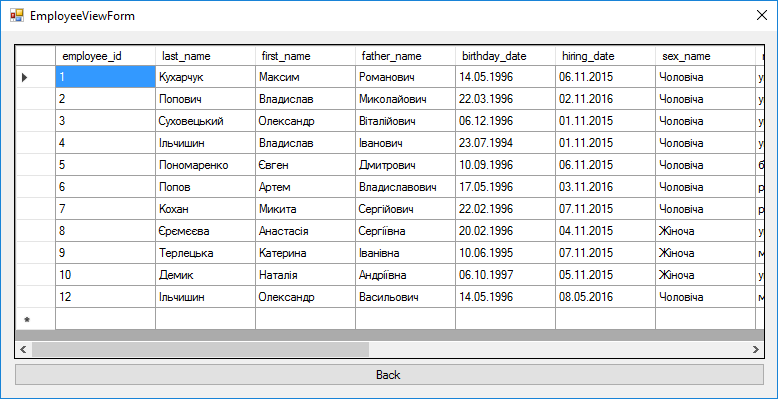


Рисунок 3.17 – Виконання запиту на вибір даних із БД

Лістинг запиту:

Select employee\_id, last\_name, first\_name, father\_name, birthday\_date, hiring\_date, sex\_name, nationality\_name, qualification\_name, phone\_number, position\_name, street\_name, house\_number, apt\_number, experience

From Employees Inner Join Sexes on Employees.sex\_fk = Sexes.sex\_id

Inner Join Nationalaties On Employees.nationality\_fk = Nationalaties.nationality\_id

Inner Join Qualifications On Employees.qualification\_fk = Qualifications.qualification\_id

Inner Join Positions On Employees.position\_fk = Positions.position\_id

Inner Join Streets On Employees.street\_fk = Streets.street\_id

* Запити на занесення даних в БД – рисунок 3.18;

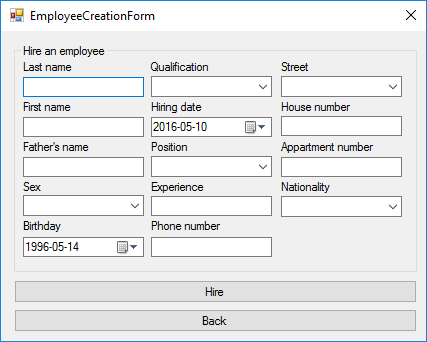


Рисунок 3.18 – Виконання запиту на занесення даних у БД

Лістинг запиту:

INSERT INTO Employees

VALUES (@last\_name, @first\_name, @father\_name, @birthday, @hiring\_date, @sex, @nat, @qual, @phone, @pos, @street, @house, @apt, @exp)

3.2.3. Розробка інтерфейсу для введення і перегляду даних

Форма– створений користувачем графічний інтерфейс із використанням елементів керування для наочного введення даних у базу. Використаються для введення нових записів (рядків таблиць), перегляду й редагування вже наявних даних, завдання параметрів запитів і виводу відповідей на них й інше. Форми являють собою прямокутні вікна з розміщеними в них елементами керування. Існує можливість створення форм динамічно при виконанні програми, однак природним режимом їхнього створення є режим візуальне конструювання.

Форми можуть містити на собі різноманітні елементи керування та графічні елементи: кнопки, вкладки, малюнки, написи, діаграми, списки, прапорці, перемикачі та інше.

Отже, форми – це зручний спосіб доступу до даних в таблицях, що забезпечує також можливість керування даними: їх видаленням, редагуванням та створенням.

Форми створюються за допомогою інтуітивного графічного інтерфейсу середовища розробки Visual Studio. Для розробки застуснку було використано засоби Windows Forms Application та мову програмування С#. Під час розробки було створено 21 форм для занесення інформації у БД – рисунок 3.19, 11 форм для перегляду інформації з БД – рисунок 3.20, 1 форму для відображення інформації про розробника – рисунок 3.21, 2 форми головного меню програми – рисунок 3.22.

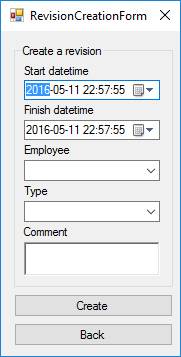


Рисунок 3.19 – Форма занесення інформації у БД

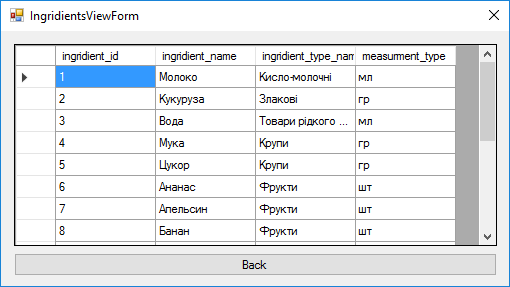


Рисунок 3.20 – Форма перегляду інформації з БД

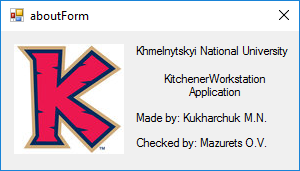


Рисунок 3.21 – Форма інформації про автора

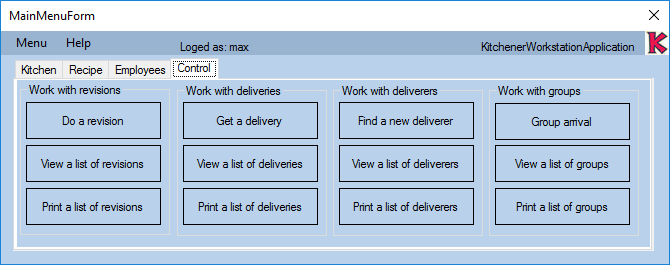


Рисунок 3.22 – Форма головного меню застосунку

3.3. Тестування інформаційної системи

Для перевірки можливостей застосунку було проведено його функціональне тестування. Згідно поставленого завдання, інформаційна система має наступні групи функцій:

* Кухня – робота з базою приготування страв.
* Рецептури – робота з базою страв, складових, інгрідієнтів.
* Контроль – робота з базою ревізій, поставок, постачальників, партій, рухів.
* Кадри – робота з базою працівників.

Відповідно до кожної групи функцій у застосунку було створено окрему вкладку за допомогою елементу TabControl. Всі групи поділені на функціональні блоки, яки виділені за допомогою GroupBox. Кожна кнопка у GroupBox виконує свою функцію, як це було описано у пункті 3.1.

Група функцій «Кухня» включає в себе такі функції:

* Створення нової картки страви – рисунок 3.23.

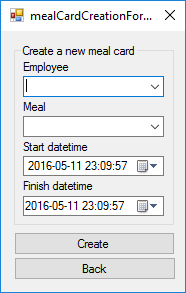
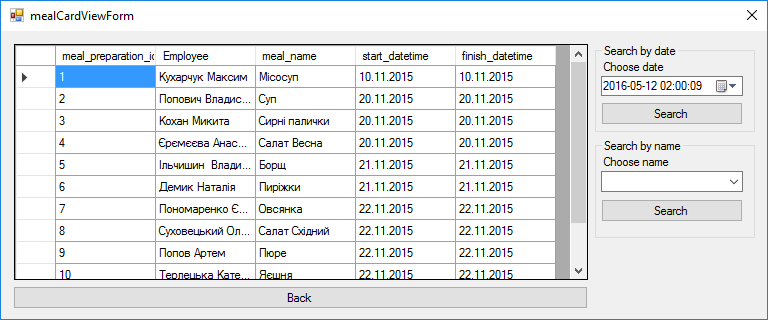


Рисунок 3.23 – Форма створення нової картки страви

* Перегляд списку карток страв з функцією параметричного пошуку – рисунок 3.24.

Рисунок 3.24 – Форма перегляду карток страв

* Вивід на друк списку карток страв – рисунок 3.25.

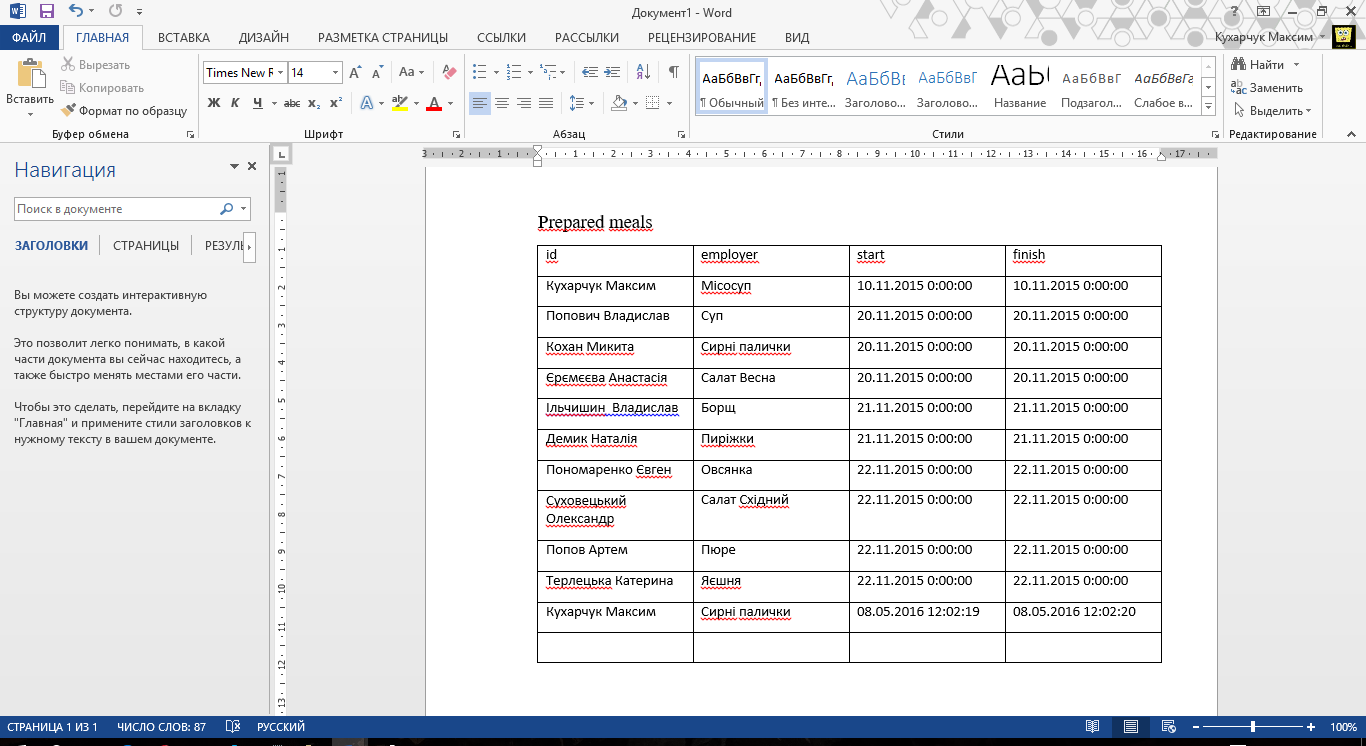


Рисунок 3.25 – Вивід на друк карток страв

Група функцій «Рецептури» включає в себе такі функції:

* Додавання нового інгрідієнту – рисунок 3.26.

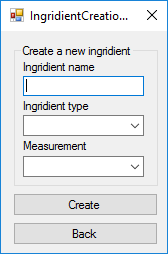
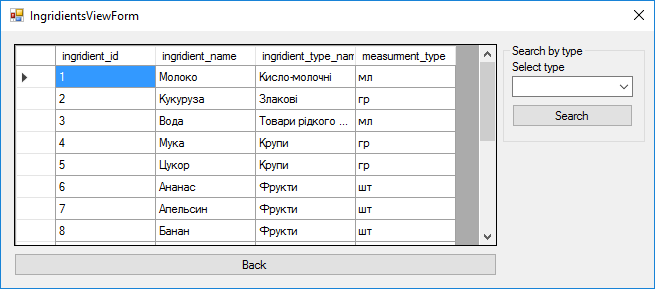


Рисунок 3.26 – Форма додавння нового інгрідієнту

* Перегляд списку інгрідієнтів з функцією параметричного пошуку – рисунок 3.27.

Рисунок 3.27 – Форма перегляду списку інгрідієнтів

* Вивід на друк списку інгрідієнтів – рисунок 3.28.

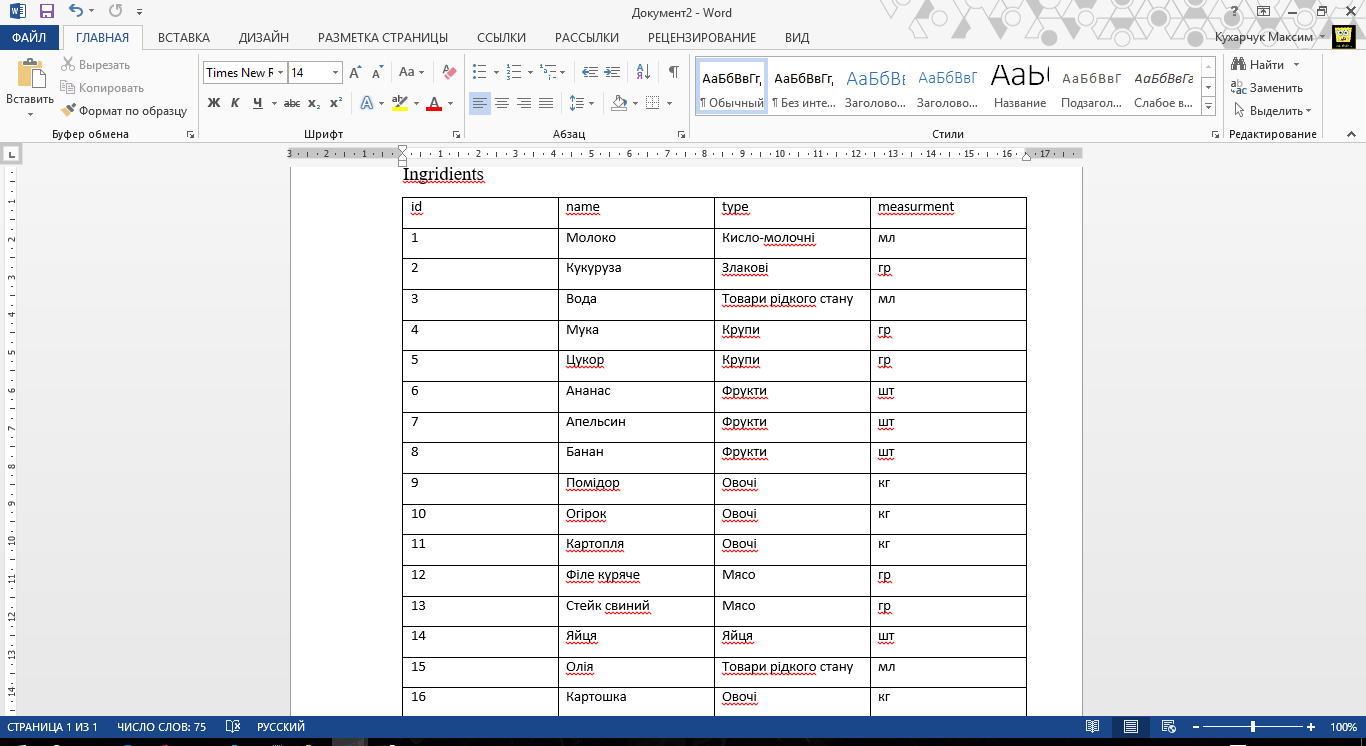


Рисунок 3.28 – Вивід на друк списку інгрідієнтів

* Додавання нової страви – рисунок 3.29.

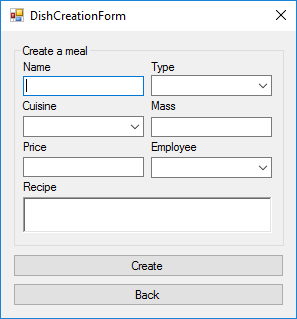
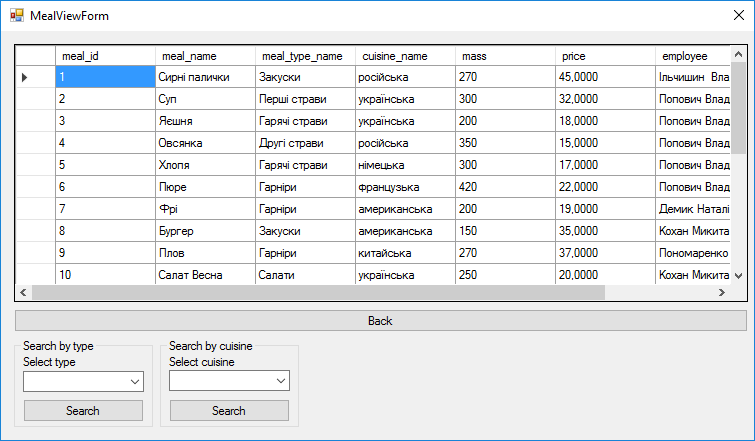


Рисунок 3.29 – Форма додавння нової страви

* Перегляд списку страв з функцією параметричного пошуку – рисунок 3.30.

Рисунок 3.30 – Форма перегляду списку страв

* Вивід на друк списку страв – рисунок 3.31.

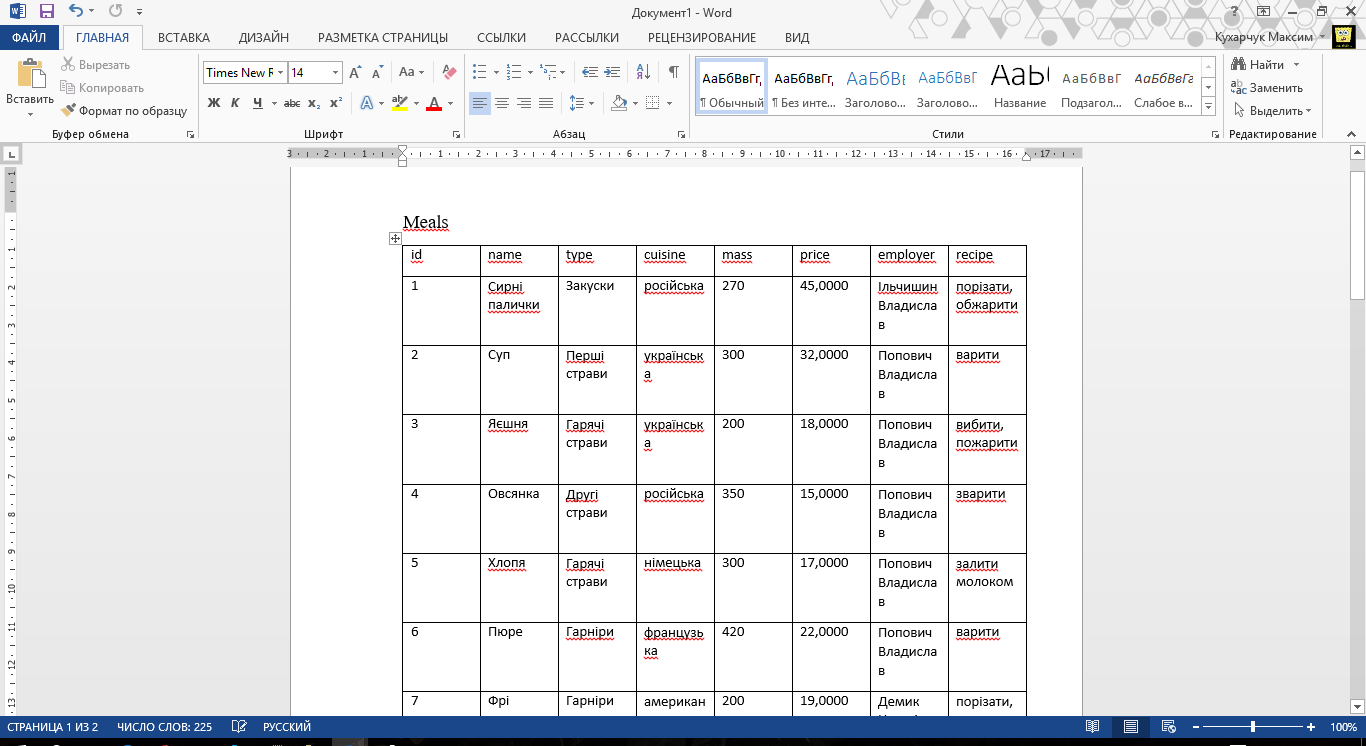


Рисунок 3.31 – Вивід на друк списку страв

* Додавання нового компоненту – рисунок 3.32.

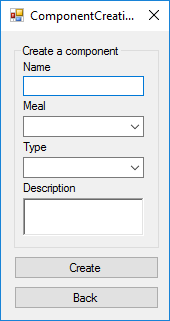
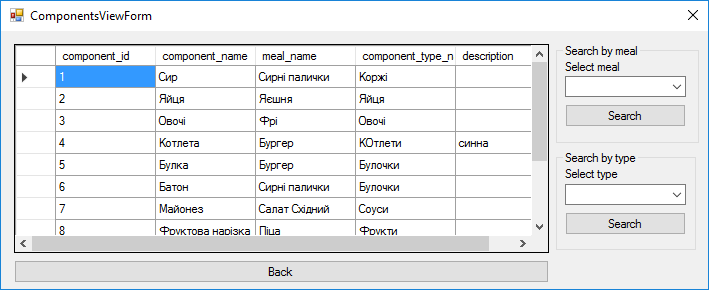


Рисунок 3.32 – Форма додавння нового компоненту

* Перегляд списку компонентів з функцією параметричного пошуку – рисунок 3.33.

Рисунок 3.33 – Форма перегляду списку компонентів

* Вивід на друк списку компонентів – рисунок 3.34.

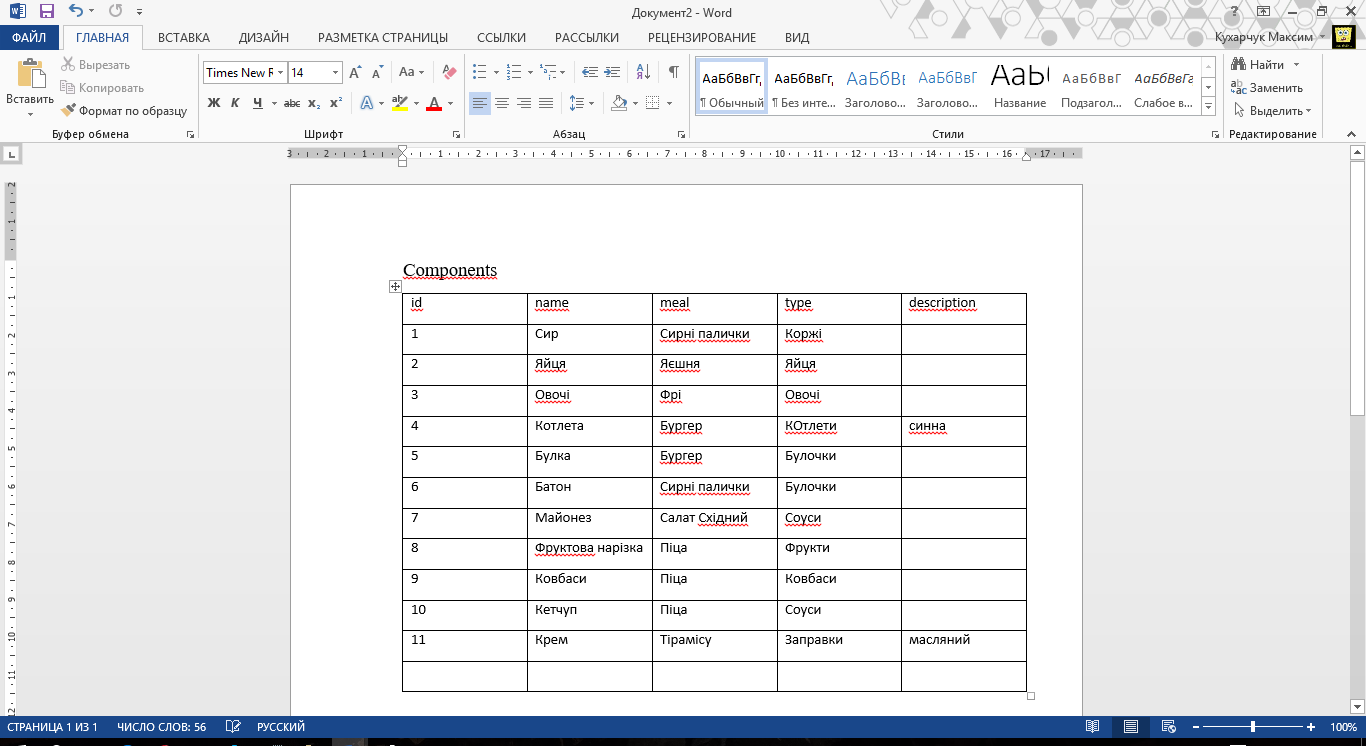


Рисунок 3.34 – Вивід на друк списку компонентів

Група функцій «Працівники» включає в себе такі функції:

* Додавання нового працівника – рисунок 3.35.

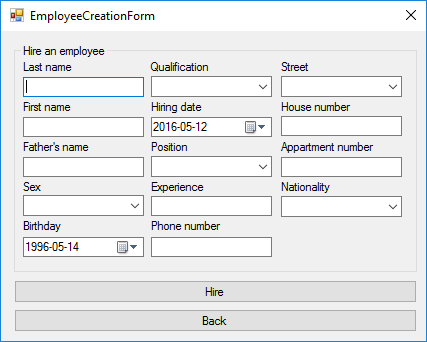
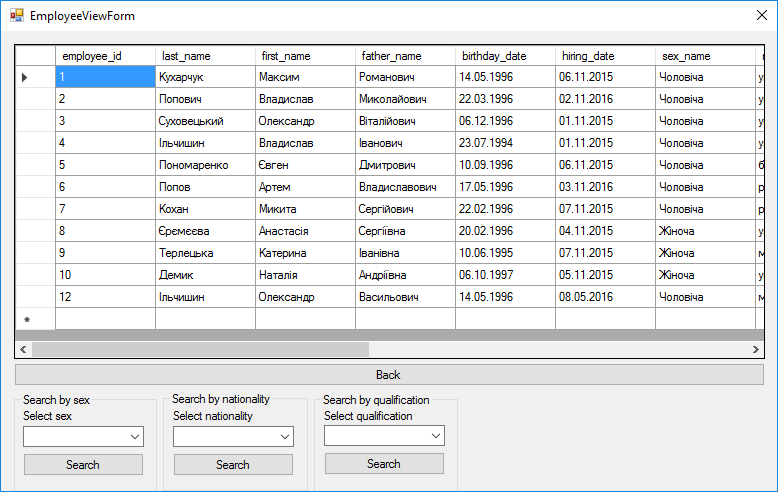


Рисунок 3.35 – Форма додавання нового працівника

* Перегляд списку працівників з функцією параметричного пошуку – рисунок 3.35.

Рисунок 3.35 – Форма перегляду списку працівників

* Вивід на друк списку працівників – рисунок 3.36.

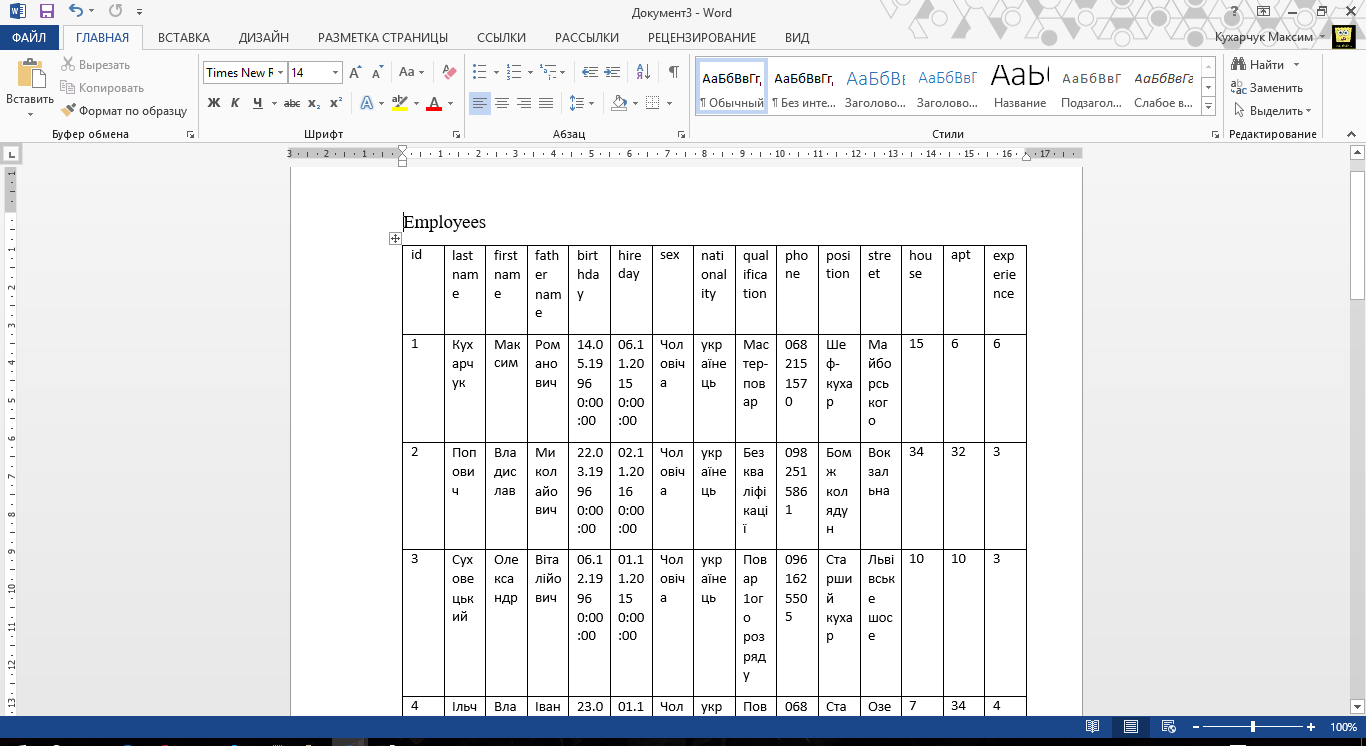


Рисунок 3.36 – Вивід на друк списку працівників

* Додавання нової посади – рисунок 3.37.

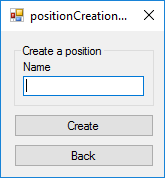


Рисунок 3.37 – Форма додавання нової посади

* Перегляд списку посад – рисунок 3.38.

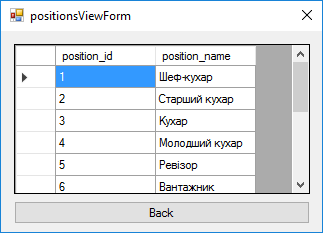


Рисунок 3.38 – Форма перегляду списку посад

* Вивід на друк списку посад – рисунок 3.38.

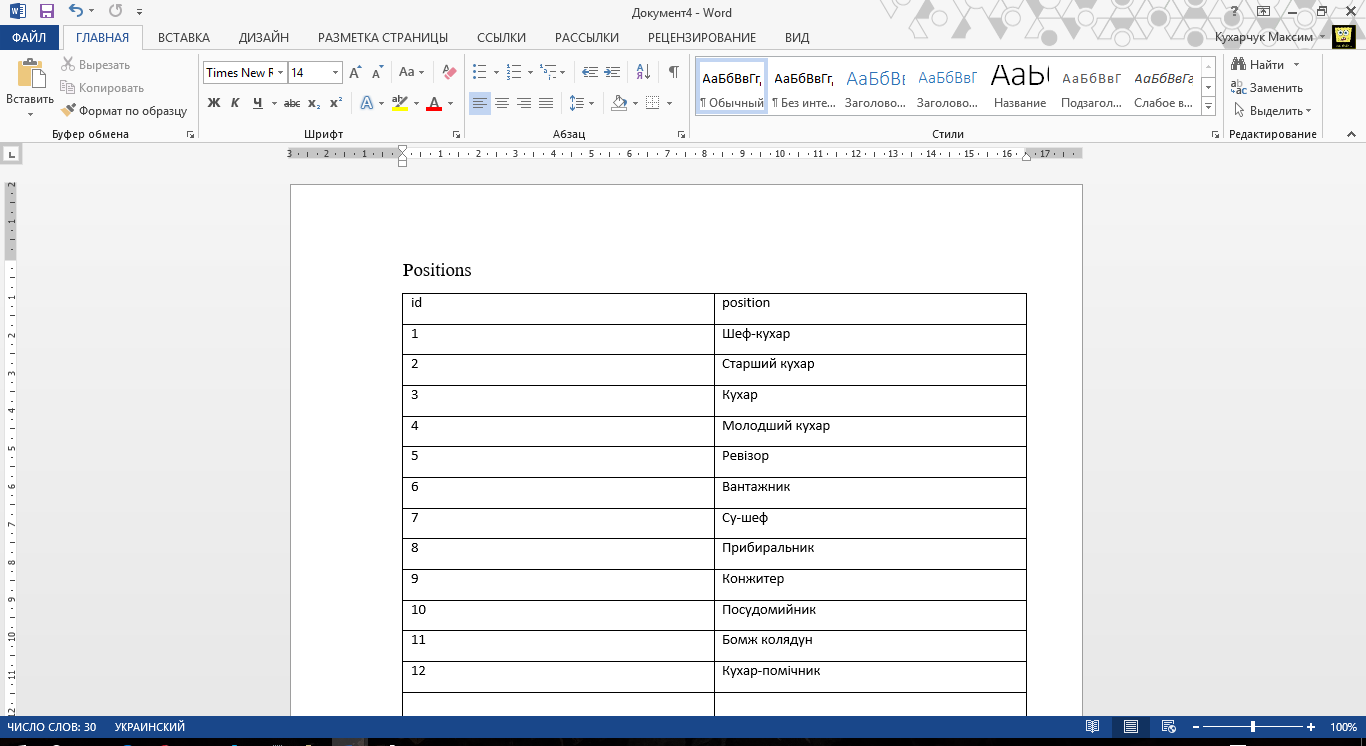


Рисунок 3.36 – Вивід на друк списку посад

* Генерування списку працівників за посадою – рисунок 3.37.

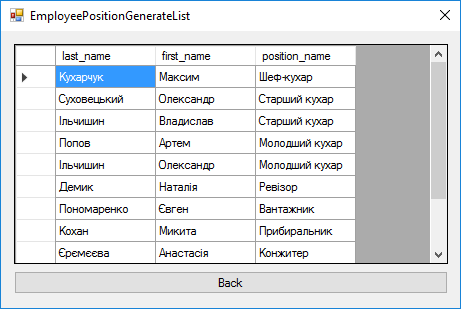


Рисунок 3.37 – Форма згенерованого списку працівників запосадами

Група функцій «Контроль» включає в себе такі функції:

* Проведення ревізії – рисунок 3.38.

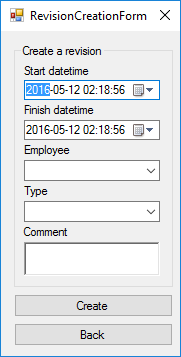
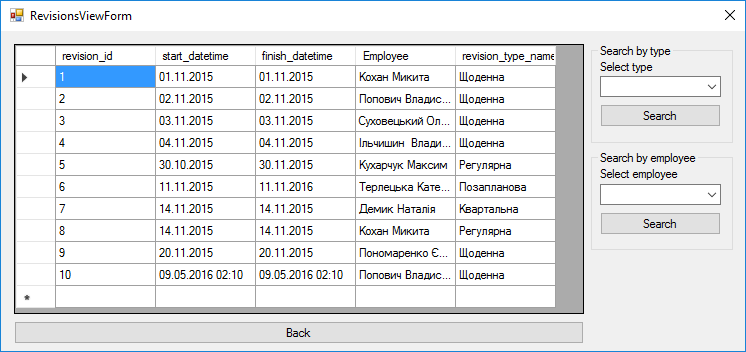


Рисунок 3.38 – Форма додавання нової ревізії

* Перегляд списку ревізій з функцією параметричного пошуку – рисунок 3.39.

Рисунок 3.39 – Форма перегляду списку ревізій

* Вивід на друк списку ревізій – рисунок 3.39.

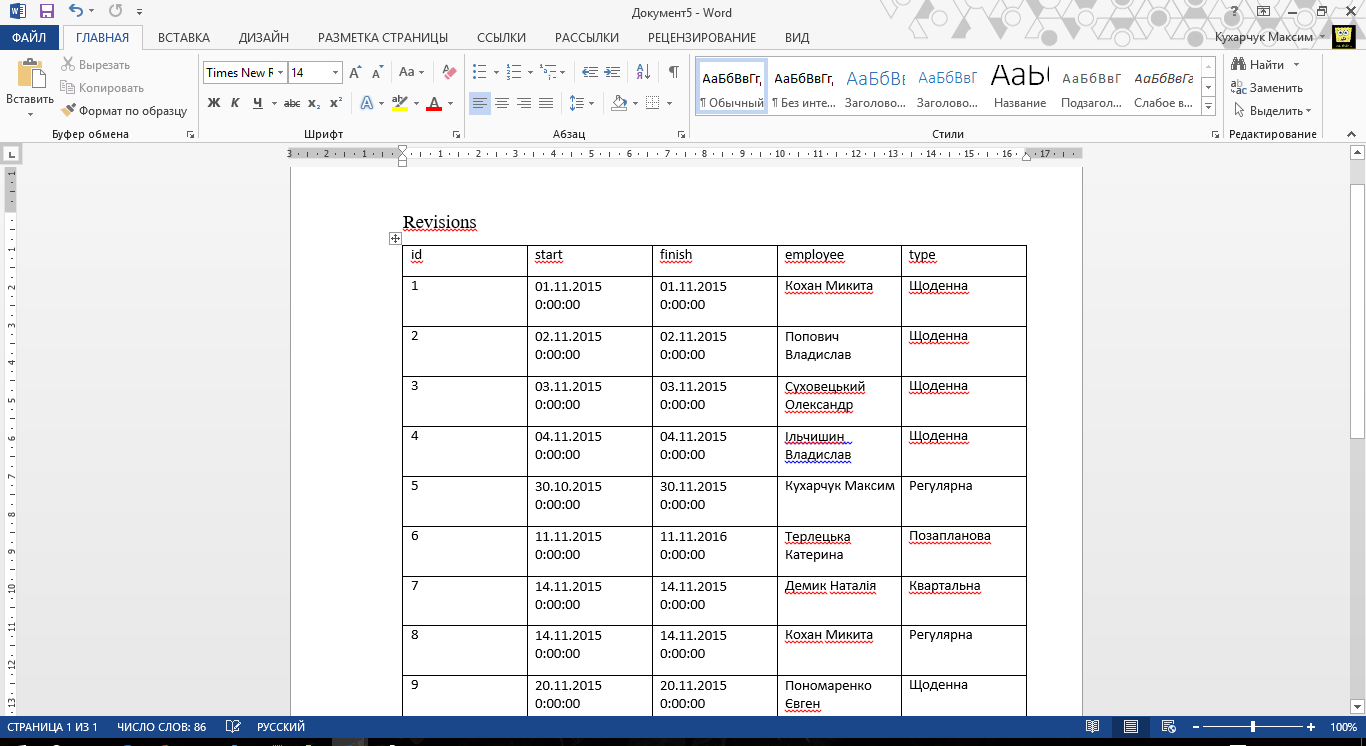


Рисунок 3.39 – Вивід на друк списку ревізій

* Отримання поставки – рисунок 3.40.

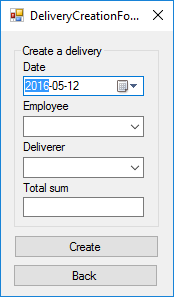
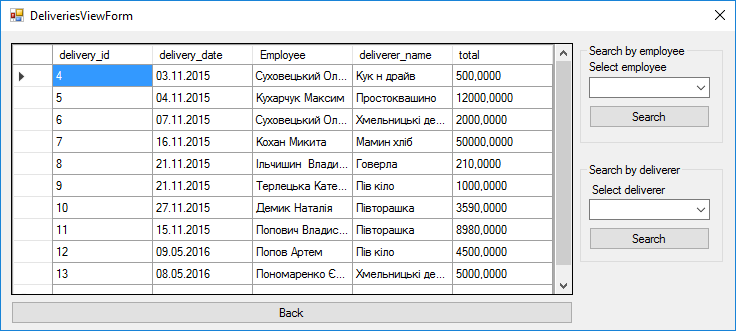


Рисунок 3.40 – Форма додавання нової поставки

* Перегляд списку поставок з функцією параметричного пошуку – рисунок 3.41.

Рисунок 3.41 – Форма перегляду списку поставок

* Вивід на друк списку поставок – рисунок 3.42.

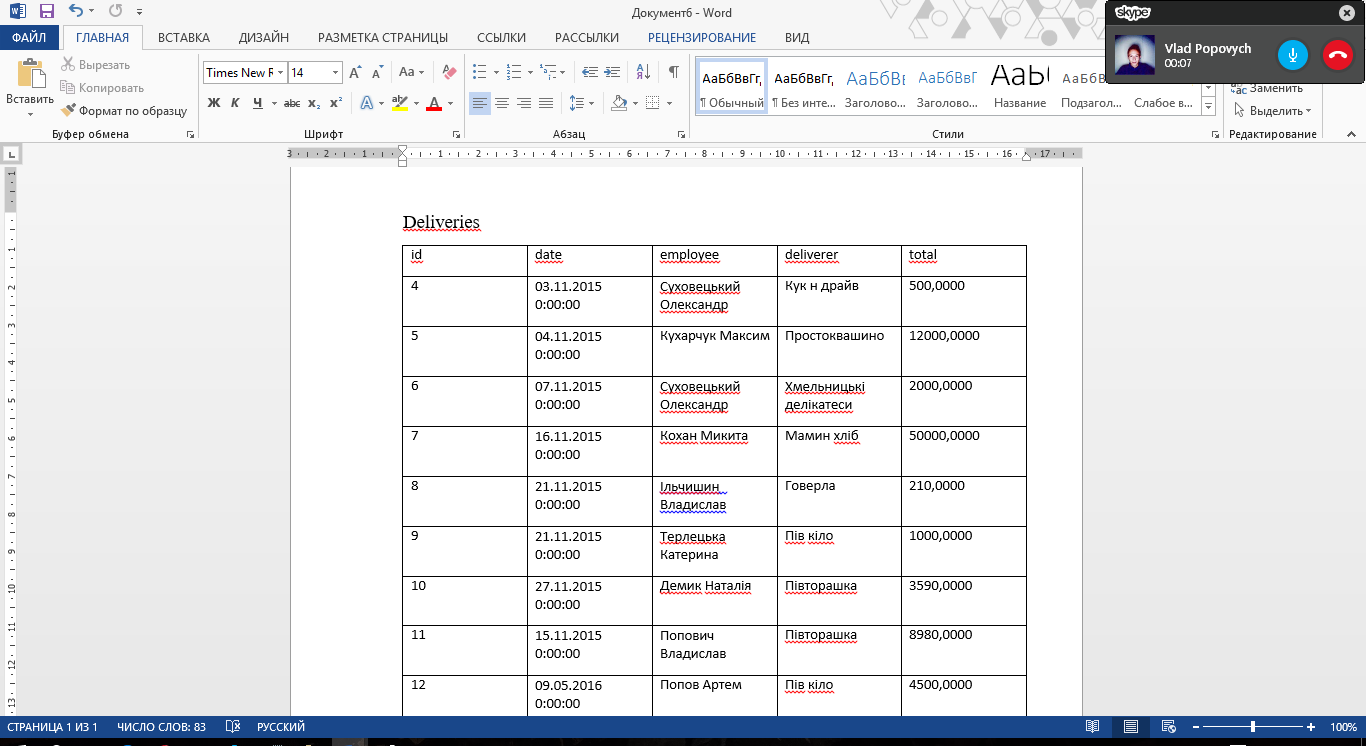


Рисунок 3.42 – Вивід на друк списку поставок

* Додавання нового постачальника – рисунок 3.43

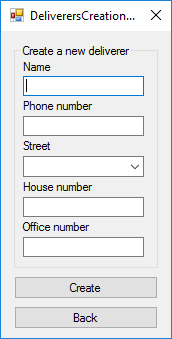
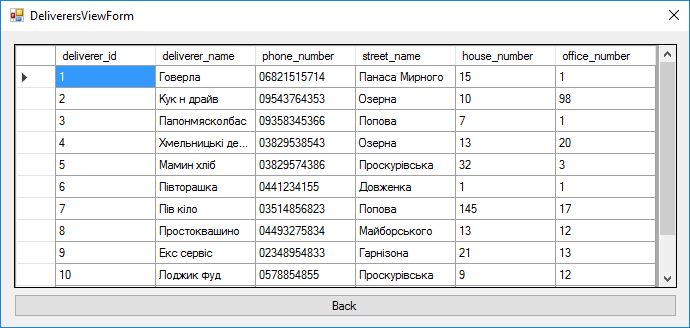


Рисунок 3.43 – Форма додавання нового постачальника

* Перегляд списку постачальників – рисунок 3.44.

Рисунок 3.44 – Форма перегляду списку постачальників

* Вивід на друк списку постачальників – рисунок 3.45.

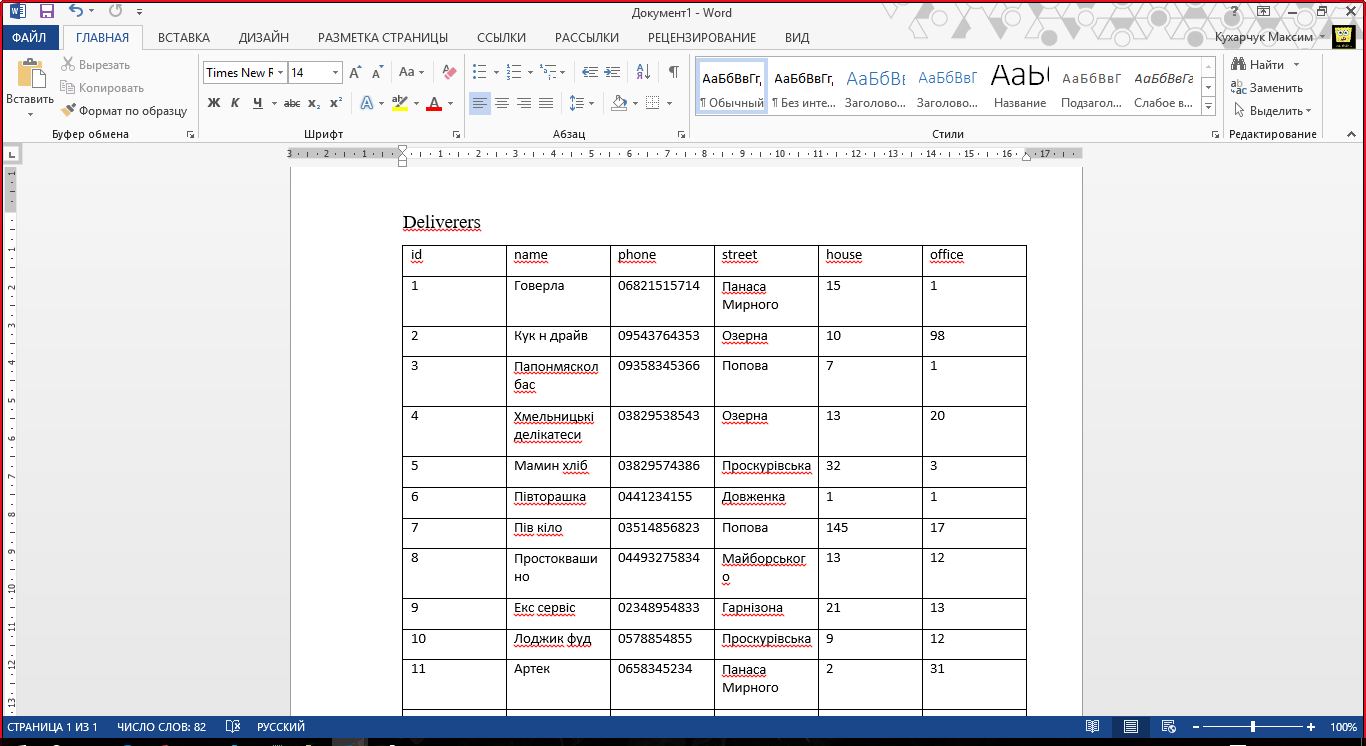


Рисунок 3.45 – Вивід на друк списку постачальників

* Додавання нової партії– рисунок 3.46

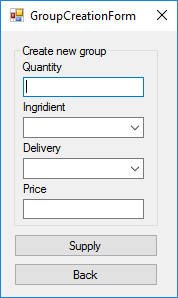
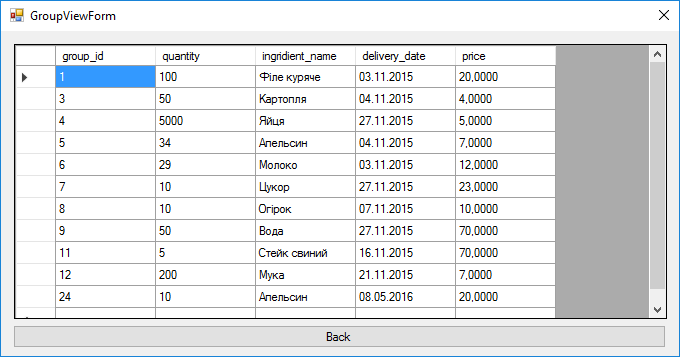


Рисунок 3.46 – Форма додавання нової партії

* Перегляд списку партій – рисунок 3.47.

Рисунок 3.47 – Форма перегляду списку партій

* Вивід на друк списку партій – рисунок 3.48.

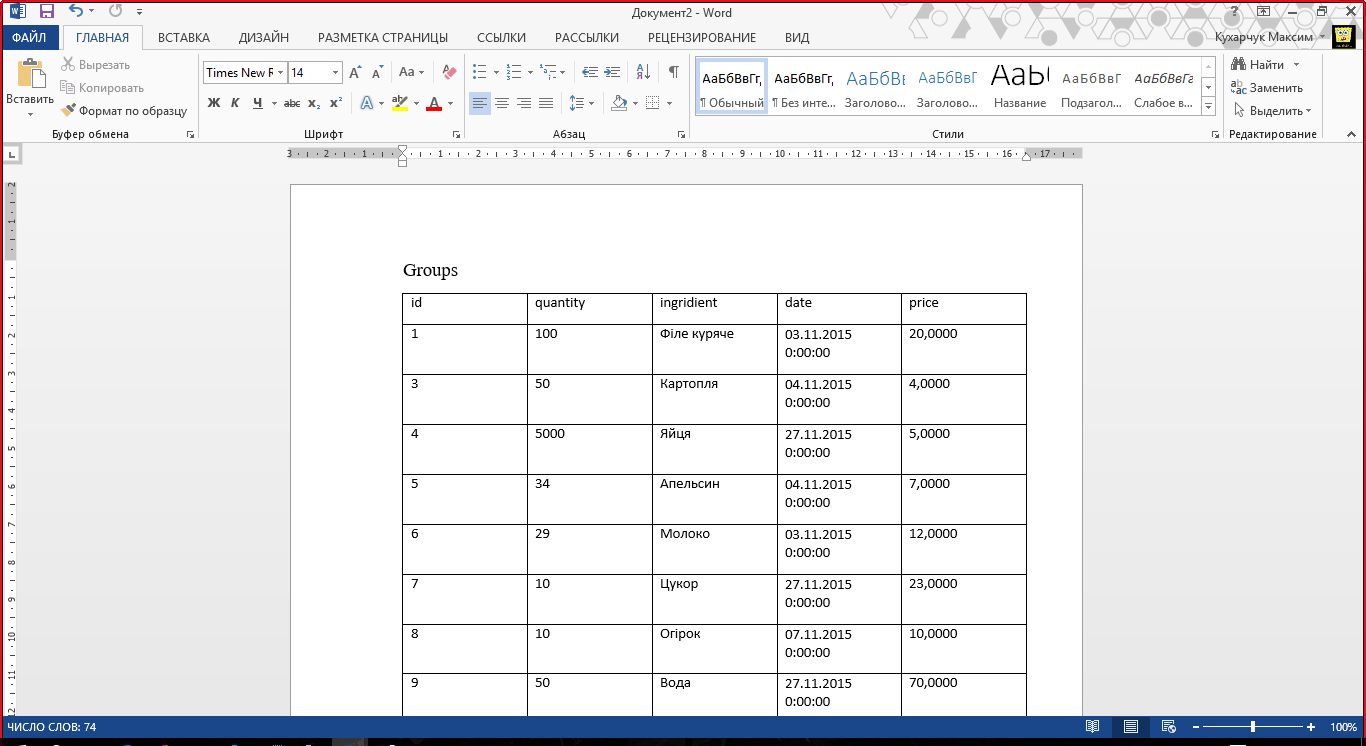


Рисунок 3.48 – Вивід на друк списку партій

Таким чином, тестування показало, що розроблюваний програмний продукт виконує всі поставлені завдання.

3.4. Інструкція користувача

Кнопкова форма – форма, єдина мета якої перенаправити користувачів до інших форм (зазвичай при клацанні мишею кнопки форми). Вона – свого роду головне меню вашої БД. Така форма – одночасно і відправна точка, і центр дій. Типова кнопкова форма містить набір кнопок, напрямних в різні місця.

Для входу у застосунок потрібно авторизуватися. Облікові записи користувачів зберігаються у БД та можуть бути редаговані тільки розробником застосунку – рисунок 3.49.

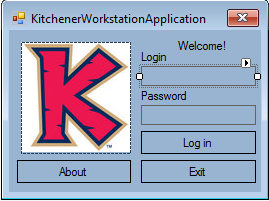


Рисунок 3.49 – Вікно авторизації засосунку

В застосунку «АРМ Кухаря» реалізовано одну головну форму, яка дає доступ до всіх даних, що зберігається в базі. Для зручності функціонал застосунку розділено за допомогою TabControl згідно до груп функцій. В ньому містяться наступні вкладки: «Кухня», «Рецептури», «Кадри», «Контроль» - рисунок 3.50.

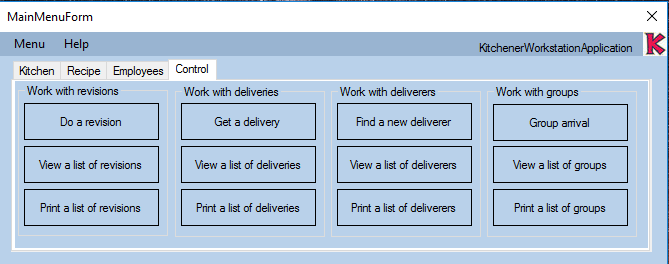


Рисунок 3.50 – Головне вікно застосунку

У кожній вкладці, групи функцій розділені по функціональним блоками та виделіне за допомогою GroupBox. Кожна кнопка викликає окрему форму для роботи з відповідною функцією. Для додавання інформації в БД викликається форма, у яку заноситься інформація. За допомогою встроєнного запиту інформація обробляється та заноситься у БД та їй присвоюється унікальний ID – рисунок 3.51.

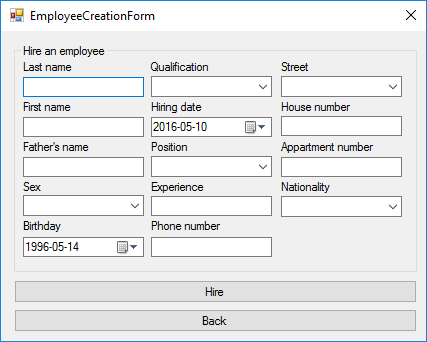


Рисунок 3.51 – Занесення інформації у БД

Для виведення інформації на екран, використовуються ті ж самі об’єкти. Відкривається форма, за допомогою запиту вибирається тільки та інформація, яка потрібна користувачеві. Відтворення інформації на екрані відбувається завдяки занесенню інформації у DataSet (міні-БД застосунку) та подальше занесення у DataGridView(елемент для роботи з таблицями). На виході користувач отримує форму, у якій зображена таблиця з потрібною інформацією – рисунок 3.52.

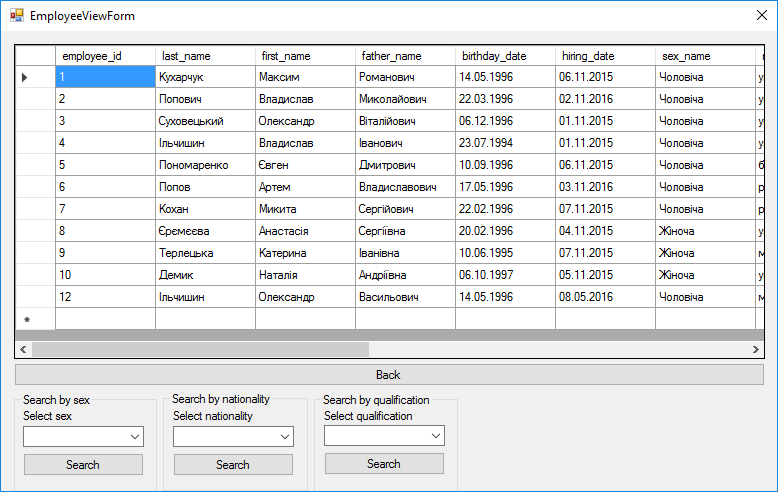


Рисунок 3.52 – Відтворення інформації з БД у застосунку

Для створення звіту на основі інформації з БД вибрано занесення її у текстовий документ у формі таблиці та подальший друк – рисунок 3.53

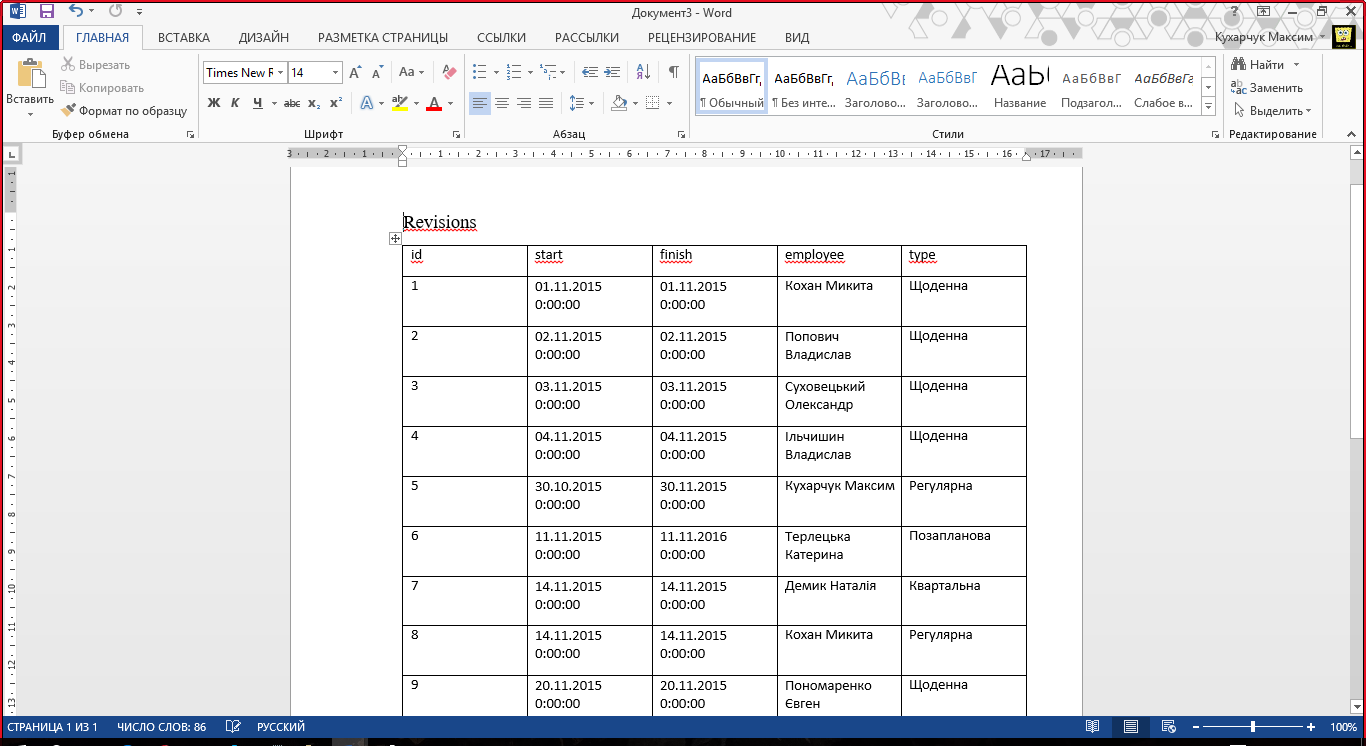


Рисунок 3.53 – Створення звіту на основі інформації з БД

Крім того реалізовано сортування інформації та вибір інформації за відповідними критеріями – рисунок 3.54. На рисунку зображено формування списку працівників за посадами. Відповідно на першому місці людина, яка займає найвищу посаду.

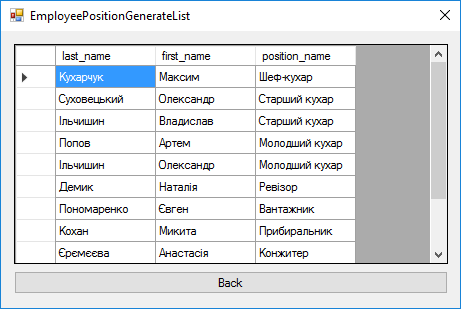


Рисунок 3.54 – Формування інформації за критеріями

Для виклику інформації про розробника створено відповідну форму – рисунок 3.55.

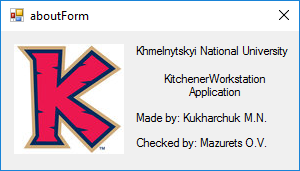


Рисунок 3.55 – Інформація про розробника

Для виходу з облікового запису або виходу з застосунку створено menuStrip з відповідними елементами – рисунок 3.56.

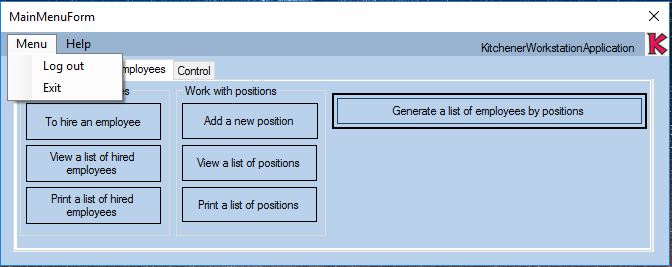


Рисунок 3.56 – Вихід з застосунку

3.5. Вимоги до програмних та технічних засобів

Мінімальні вимоги до обладнання та програмного забезпечення для коректної роботи застосунку приведені у таблиці.

|  |  |
| --- | --- |
| Компонент | Мінімальні вимоги |
| ПК та процесор | Процесор з частотою 1000 МГц |
| Пам’ять | 1024 МБ оперативної пам'яті |
| Жорсткий диск | 5 ГБ вільного дискового простору |
| Монітор | Монітор з роздільною здатністю  1024х768 |

Щодо програмного забезпечення, то підтримуються тільки 64-розрядні версії: Windows 7, Windows 8, Windows 8.1, Windows 10. Не підтримуються версії: 32-розрядна версія Windows 7, 32 та 64-розрядні версії Windows XP та нижче.

Висновки

Отже, в рамках виконання курсового проекту, було створено застосунок «АРМ Кухаря». Згідно до предметної області було розроблено структуру БД, що дозволяє зберігати усю потрібну інформацію.

Розроблювана система зберігає каталоги предметної області (досліджені й наведені вище): страви, складові, інгрідієнти, ревізії, поставки тощо. Застосування такої системи покликане оптимізувати роботу кухарів. Робота з застосунком не потребує високих системних вимог комп’ютера та програмного забезпечення. Усе, що потрібно для роботи з нею – це встановлений Microsoft SQL Server 2014, 64-розрядна операційна система не нижче Windows 7 та Net Framework версії 4.5.

Реалізовано графічний інтерфейс, головне меню якого розділено на вкладки відповідно до груп функцій. Функції було реалізовано повністю відповідно до поставленного завдання.

Розроблений програмний продукт може застосовуватись кухарями для автоматизації їх робочого місця, адже аналогів даного ПП на ринку не існує.

Перелік посилань

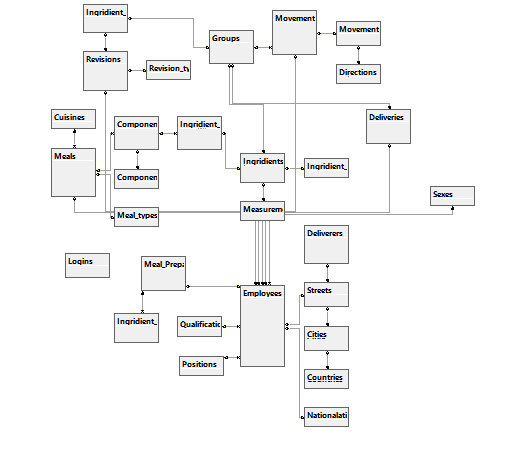
* 1. Автоматизоване робоче місце [Електронний ресурс]. Режим доступу: https://ru.wikipedia.org/wiki/Автоматизированное\_рабочее\_место
  2. Підприємство громадського харчування [Електронний ресурс]. Режим доступу: http://lubbook.net/book\_317\_glava\_4\_2.html
  3. Кухар [Електронний ресурс]. Режим доступу: https://uk.wikipedia.org/wiki/%D0%9A%D1%83%D1%85%D0%B0%D1%80
  4. Microsoft Office [Електронний ресурс]. Режим доступу: https://products.office.com/uk-ua/home
  5. Microsoft Word [Електронний ресурс]. Режим доступу: https://ru.wikipedia.org/wiki/Microsoft\_Word

1. Microsoft Excel [Електронний ресурс]. Режим доступу: https://ru.wikipedia.org/wiki/Microsoft\_Excel
2. iОбщепит [Електронний ресурс]. Режим доступу: http://iobshepit.com/obzor/
3. АРМ Повара PlayMarket [Електронний ресурс]. Режим доступу: https://play.google.com/store/apps/details?id=ru.rarus.mobilechef&hl=ru
4. Net [Електронний ресурс]. Режим доступу: https://ru.wikipedia.org/wiki/.NET\_Framework
5. Java [Електронний ресурс]. Режим доступу: https://ru.wikipedia.org/wiki/Java
6. C++ [Електронний ресурс]. Режим доступу: http://uk.wikipedia.org/wiki/C%2B%2B.
7. C# [Електронний ресурс]. Режим доступу: http://uk.wikipedia.org/wiki/C\_Sharp
8. Sql [Електронний ресурс]. Режим доступу: https://uk.wikipedia.org/wiki/Microsoft\_SQL\_Server.
9. Access [Електронний ресурс]. Режим доступу: https://uk.wikipedia.org/wiki/Microsoft\_Access.
10. access запити таблиці[Електронний ресурс]. Режим доступу: http://ua.textreferat.com/referat-7616.html
11. Oracle [Електронний ресурс]. Режим доступу: <http://uk.wikipedia.org/wiki/Oracle_Database>.
12. Cтрава [Електронний ресурс]. Режим доступу: <https://uk.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D0%B0%D0%B2%D0%B0>

**Додатки**

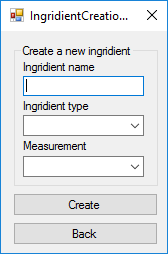
**Додаток А**

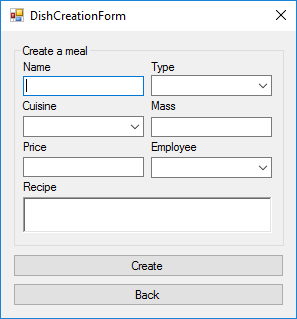
**Даталогічна модель БД**



**Додаток Б**

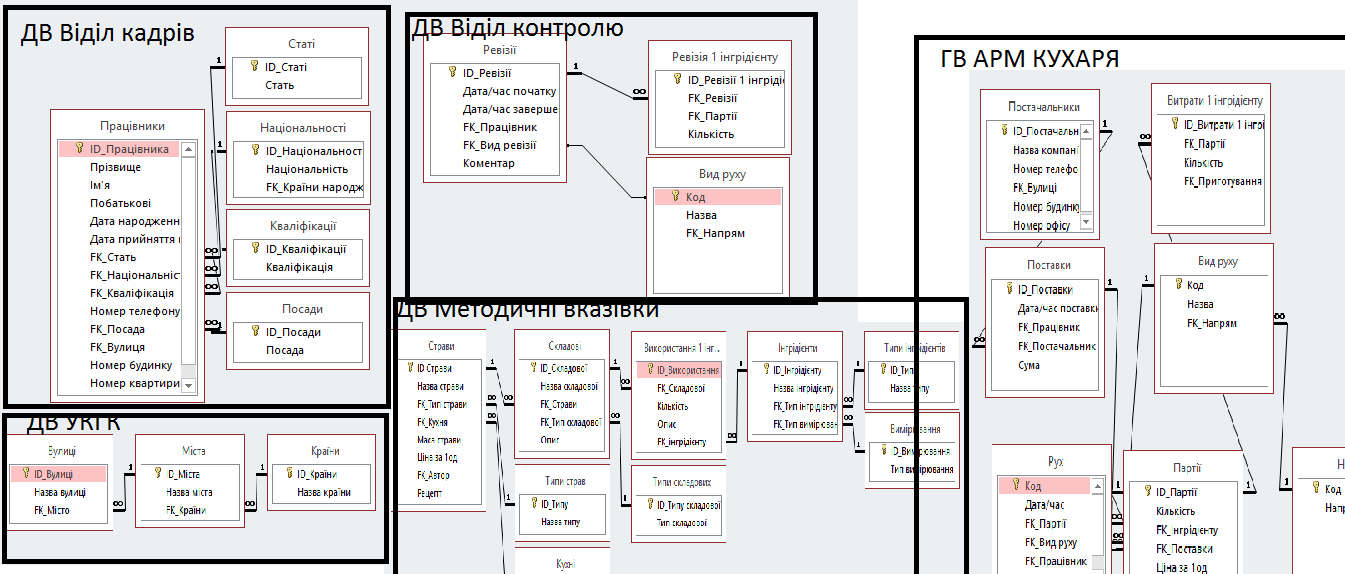
**Приклади оформлення вхідної інформації**





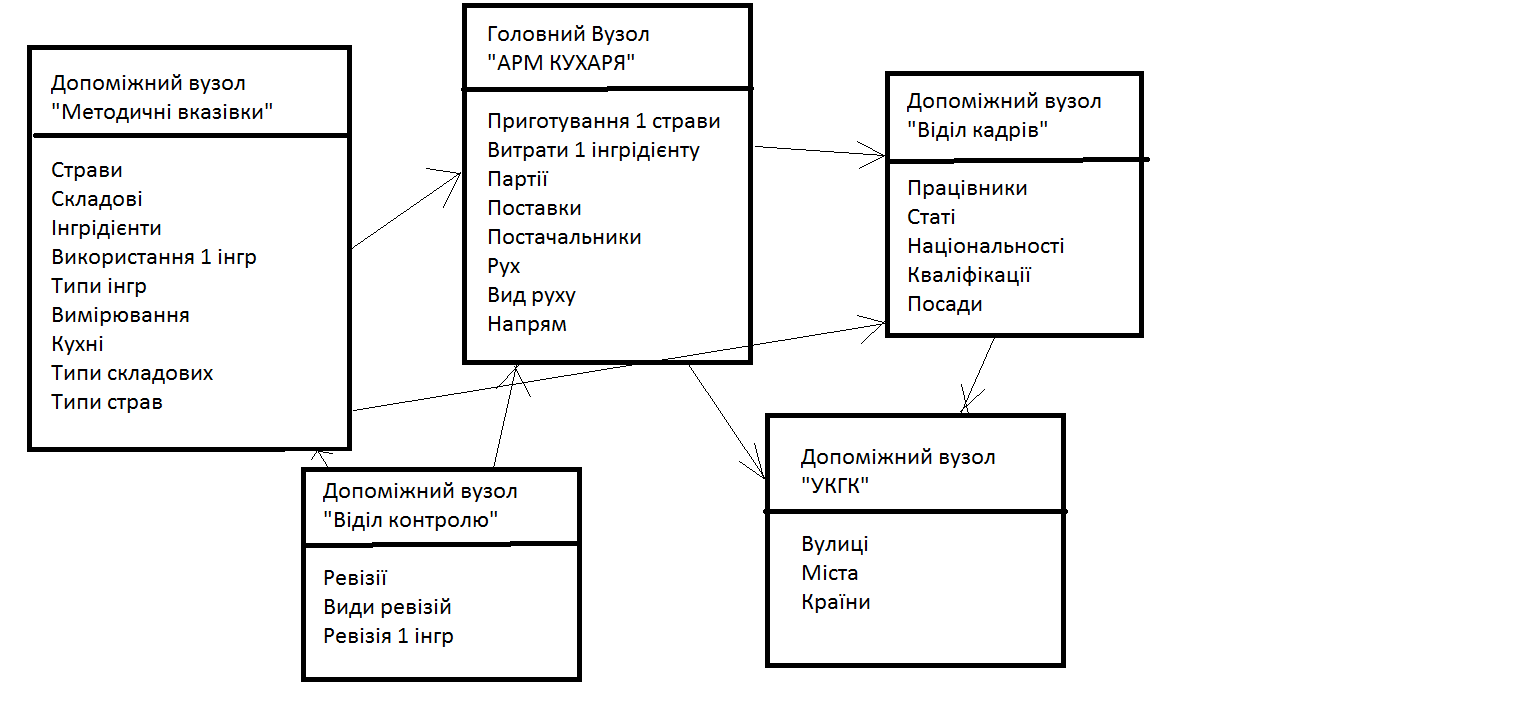
**Додаток В**

**Даталогічна модель розподіленої бази даних**



**Додаток Г**

**Вузлова схема розподіленої бази даних**



**Додаток Д**

**Список імен вузлів і таблиць БД**

|  |  |
| --- | --- |
| **Методичні вказівки** | **metoduchni\_vkazivku** |
| Страви | stravu |
| Складові | skladovi |
| Інгрідієнти | ingridientu |
| Вик. 1 інгр. | vuk\_1\_ingr |
| Типи інгр. | tupu\_1\_ingr |
| Вимірювання | vumiryuvannya |
| Кухні | kuhni |
| Типи складових | typi\_skladovuh |
| Типи страв | typi\_strav |
| **УГК** | **ukgk** |
| Вулиці | vyluci |
| Міста | mista |
| Країни | krainu |
| **Відділ кадрів** | **vidil\_kadriv** |
| Працівники | pracivnyku |
| Статі | stati |
| Національності | nacionalnosti |
| Кваліфікації | kvalifikacii |
| Посади | posadu |
| **Віділ контролю** | **vidil\_kontrolyu** |
| Ревізії | revizii |
| Види ревізій | vudu\_reviziy |
| Ревізії 1 інгр | revizii\_1\_ingr |

.

**Додаток Е**

**Каталог системних табличних імен**

kukharchuk@home.stravu@metoduchni\_vkazivku

kukharchuk@home.skladovi@metoduchni\_vkazivku

kukharchuk@home.ingridientu@metoduchni\_vkazivku

kukharchuk@home.vuk\_1\_ingr@metoduchni\_vkazivku

kukharchuk@home.tipi\_ingr@metoduchni\_vkazivku

kukharchuk@home.vumiryuvannya@metoduchni\_vkazivku

kukharchuk@home.kuhni@metoduchni\_vkazivku

kukharchuk@home.tupu\_skladovuh@metoduchni\_vkazivku

kukharchuk@home.tupu\_strav@metoduchni\_vkazivku

kukharchuk@home.vyluci@ukgk

kukharchuk@home.vyluci@ukgk

kukharchuk@home.krainu@ukgk

kukharchuk@home.pracivnyku@vidil\_kadriv

kukharchuk@home.stati@vidil\_kadriv

kukharchuk@home.nacionalnosti@vidil\_kadriv

kukharchuk@home.kvalifikacii@vidil\_kadriv

kukharchuk@home.posadu@vidil\_kadriv

kukharchuk@home.revizii@vidil\_kontrolyu

kukharchuk@home.vudu\_reviziy@vidil\_kontrolyu

kukharchuk@home.revizii\_1\_ingr@vidil\_kontrolyu

**Додаток Є**

**Програмні коди**

public partial class ComponentCreationForm : Form

{

public ComponentCreationForm()

{

InitializeComponent();

comboBox1Fill();

comboBox2Fill();

}

private void comboBox1Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT meal\_name FROM Meals";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox1.Items.Add(ds.Tables[0].Rows[i]["meal\_name"]);

}

}

private void comboBox2Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT component\_type\_name FROM Component\_types";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox2.Items.Add(ds.Tables[0].Rows[i]["component\_type\_name"]);

}

}

private int meal\_get\_id(string meal\_name)

{

int id = 0;

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select meal\_id, meal\_name FROM Meals";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

object meal\_id = reader.GetValue(0);

object mealName = reader.GetValue(1);

if (meal\_name == mealName.ToString())

{

id = Convert.ToInt16(meal\_id);

}

}

return id;

}

private int componentType\_get\_id(string compType)

{

int id = 0;

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select component\_type\_id, component\_type\_name FROM Component\_types";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

object component\_type\_id = reader.GetValue(0);

object component\_type\_name = reader.GetValue(1);

if (compType == component\_type\_name.ToString())

{

id = Convert.ToInt16(component\_type\_id);

}

}

return id;

}

private int addComponent(string name, int meal, int type, string desc)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "INSERT INTO Components VALUES (@name, @meal, @type, @desc)";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlParameter nameParam = new SqlParameter("@name", name);

command.Parameters.Add(nameParam);

SqlParameter mealParam = new SqlParameter("@meal", meal);

command.Parameters.Add(mealParam);

SqlParameter typeParam = new SqlParameter("@type", type);

command.Parameters.Add(typeParam);

SqlParameter descParam = new SqlParameter("@desc", desc);

command.Parameters.Add(descParam);

int result = command.ExecuteNonQuery();

return result;

}

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

{

this.Close();

}

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

{

string name = textBox1.Text;

string meal = comboBox1.Text;

int meal\_id = meal\_get\_id(meal);

string compType = comboBox2.Text;

int compType\_id = componentType\_get\_id(compType);

string description = richTextBox1.Text;

int result = addComponent(name, meal\_id, compType\_id, description);

if (result == 1) MessageBox.Show("Success");

}

}

public partial class ComponentsViewForm : Form

{

public ComponentsViewForm()

{

InitializeComponent();

fillDataGridView();

comboBox1Fill();

comboBox2Fill();

}

private void comboBox2Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT component\_type\_name FROM Component\_types";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox2.Items.Add(ds.Tables[0].Rows[i]["component\_type\_name"]);

}

}

private void comboBox1Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT meal\_name FROM Meals";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox1.Items.Add(ds.Tables[0].Rows[i]["meal\_name"]);

}

}

private void fillDataGridView()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true;";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

string sqlExpression = "Select component\_id, component\_name, meal\_name, component\_type\_name, description From Components Inner Join Meals On Components.meal\_fk = Meals.meal\_id Inner Join Component\_types On Components.component\_type\_fk = Component\_types.component\_type\_id";

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

dataGridView1.DataSource = ds.Tables[0];

}

private void fillDataGridView2(string name)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true;";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

string sqlExpression = "Select component\_id, component\_name, meal\_name, component\_type\_name, description From Components Inner Join Meals On Components.meal\_fk = Meals.meal\_id Inner Join Component\_types On Components.component\_type\_fk = Component\_types.component\_type\_id WHERE component\_type\_name = @name";

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataAdapter adapter = new SqlDataAdapter(command);

SqlParameter nameParam = new SqlParameter("@name", name);

adapter.SelectCommand.Parameters.Add(nameParam);

DataSet ds = new DataSet();

adapter.Fill(ds);

dataGridView1.DataSource = ds.Tables[0];

}

private void fillDataGridView1(string name)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true;";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

string sqlExpression = "Select component\_id, component\_name, meal\_name, component\_type\_name, description From Components Inner Join Meals On Components.meal\_fk = Meals.meal\_id Inner Join Component\_types On Components.component\_type\_fk = Component\_types.component\_type\_id WHERE meal\_name = @name";

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataAdapter adapter = new SqlDataAdapter(command);

SqlParameter nameParam = new SqlParameter("@name", name);

adapter.SelectCommand.Parameters.Add(nameParam);

DataSet ds = new DataSet();

adapter.Fill(ds);

dataGridView1.DataSource = ds.Tables[0];

}

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

{

this.Close();

}

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

{

string name = comboBox1.Text;

fillDataGridView1(name);

}

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

{

string name = comboBox2.Text;

fillDataGridView2(name);

}

public partial class DeliverersCreationForm : Form

{

public DeliverersCreationForm()

{

InitializeComponent();

comboBox1Fill();

}

private void comboBox1Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT street\_name FROM Streets";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox1.Items.Add(ds.Tables[0].Rows[i]["street\_name"]);

}

}

private int street\_get\_id(string strName)

{

int id = 0;

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select street\_id, street\_name FROM Streets";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

object str\_id = reader.GetValue(0);

object str\_name = reader.GetValue(1);

if (strName == str\_name.ToString())

{

id = Convert.ToInt16(str\_id);

}

}

return id;

}

private int addDeliverer(string name, string phone, int street\_id, int house, int office)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "INSERT INTO Deliverers VALUES (@name, @phone, @street\_id, @house, @office)";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlParameter nameParam = new SqlParameter("@name", name);

command.Parameters.Add(nameParam);

SqlParameter phoneParam = new SqlParameter("@phone", phone);

command.Parameters.Add(phoneParam);

SqlParameter streetParam = new SqlParameter("@street\_id", street\_id);

command.Parameters.Add(streetParam);

SqlParameter houseParam = new SqlParameter("@house", house);

command.Parameters.Add(houseParam);

SqlParameter officeParam = new SqlParameter("@office", office);

command.Parameters.Add(officeParam);

int result = command.ExecuteNonQuery();

return result;

}

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

{

this.Close();

}

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

{

string name = textBox1.Text;

string phone = textBox2.Text;

string street = comboBox1.Text;

int street\_id = street\_get\_id(street);

int house = int.Parse(textBox3.Text);

int office = int.Parse(textBox4.Text);

int result = addDeliverer(name, phone, street\_id, house, office);

if (result == 1) MessageBox.Show("Success");

}

}

public partial class DeliverersViewForm : Form

{

public DeliverersViewForm()

{

InitializeComponent();

fillDataGridView();

}

private void fillDataGridView()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true;";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

string sqlExpression = "Select deliverer\_id,deliverer\_name, phone\_number, street\_name, house\_number, office\_number From Deliverers Inner Join Streets On Deliverers.street\_fk = Streets.street\_id";

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

dataGridView1.DataSource = ds.Tables[0];

}

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

{

this.Close();

}

}

public partial class DeliveriesViewForm : Form

{

public DeliveriesViewForm()

{

InitializeComponent();

fillDataGridView();

comboBox1Fill();

comboBox2Fill();

}

private void comboBox1Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT last\_name, first\_name FROM Employees";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox1.Items.Add(ds.Tables[0].Rows[i]["last\_name"] + " " + ds.Tables[0].Rows[i]["first\_name"]);

}

}

private void comboBox2Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT deliverer\_name FROM Deliverers";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox2.Items.Add(ds.Tables[0].Rows[i]["deliverer\_name"]);

}

}

private int employee\_get\_ID(string[] empName)

{

int id = 0;

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select employee\_id, last\_name, first\_name FROM Employees";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

object employeeID = reader.GetValue(0);

object employee\_last\_name = reader.GetValue(1);

object employee\_first\_name = reader.GetValue(2);

if (employee\_last\_name.ToString() == empName[0] && employee\_first\_name.ToString() == empName[1])

{

id = Convert.ToInt16(employeeID);

}

}

return id;

}

private void fillDataGridView()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true;";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

string sqlExpression = "Select delivery\_id, delivery\_date, last\_name + ' ' + first\_name AS Employee, deliverer\_name, total From Deliveries INNER JOIN Employees ON Deliveries.employee\_fk = Employees.employee\_id Inner Join Deliverers On Deliveries.deliverer\_fk = Deliverers.deliverer\_id";

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

dataGridView1.DataSource = ds.Tables[0];

}

private void fillDataGridView1(int id)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true;";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

string sqlExpression = "Select delivery\_id, delivery\_date, last\_name + ' ' + first\_name AS Employee, deliverer\_name, total From Deliveries INNER JOIN Employees ON Deliveries.employee\_fk = Employees.employee\_id Inner Join Deliverers On Deliveries.deliverer\_fk = Deliverers.deliverer\_id WHERE employee\_id = @id";

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataAdapter adapter = new SqlDataAdapter(command);

SqlParameter idParam = new SqlParameter("@id", id);

adapter.SelectCommand.Parameters.Add(idParam);

DataSet ds = new DataSet();

adapter.Fill(ds);

dataGridView1.DataSource = ds.Tables[0];

}

private void fillDataGridView2(string name)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true;";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

string sqlExpression = "Select delivery\_id, delivery\_date, last\_name + ' ' + first\_name AS Employee, deliverer\_name, total From Deliveries INNER JOIN Employees ON Deliveries.employee\_fk = Employees.employee\_id Inner Join Deliverers On Deliveries.deliverer\_fk = Deliverers.deliverer\_id WHERE deliverer\_name = @name";

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataAdapter adapter = new SqlDataAdapter(command);

SqlParameter nameParam = new SqlParameter("@name", name);

adapter.SelectCommand.Parameters.Add(nameParam);

DataSet ds = new DataSet();

adapter.Fill(ds);

dataGridView1.DataSource = ds.Tables[0];

}

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

{

this.Close();

}

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

{

string employeeName = comboBox1.Text;

string[] employeeNameArray = employeeName.Split(new char[] { ' ' }, StringSplitOptions.RemoveEmptyEntries);

int employeeID = employee\_get\_ID(employeeNameArray);

fillDataGridView1(employeeID);

}

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

{

string name = comboBox2.Text;

fillDataGridView2(name);

}

}

public partial class DeliveryCreationForm : Form

{

public DeliveryCreationForm()

{

InitializeComponent();

comboBox1Fill();

comboBox2Fill();

}

private void comboBox1Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT last\_name, first\_name FROM Employees";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox1.Items.Add(ds.Tables[0].Rows[i]["last\_name"] + " " + ds.Tables[0].Rows[i]["first\_name"]);

}

}

private void comboBox2Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT deliverer\_name FROM Deliverers";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox2.Items.Add(ds.Tables[0].Rows[i]["deliverer\_name"]);

}

}

private int deliverer\_get\_id(string deliverer\_name)

{

int id = 0;

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select deliverer\_id, deliverer\_name FROM Deliverers";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

object dev\_id = reader.GetValue(0);

object dev\_name = reader.GetValue(1);

if (deliverer\_name == dev\_name.ToString())

{

id = Convert.ToInt16(dev\_id);

}

}

return id;

}

private int employee\_get\_ID(string[] empName)

{

int id = 0;

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select employee\_id, last\_name, first\_name FROM Employees";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

object employeeID = reader.GetValue(0);

object employee\_last\_name = reader.GetValue(1);

object employee\_first\_name = reader.GetValue(2);

if (employee\_last\_name.ToString() == empName[0] && employee\_first\_name.ToString() == empName[1])

{

id = Convert.ToInt16(employeeID);

}

}

return id;

}

private int addDelivery(string date, int emp, int deliverer, int total)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "INSERT INTO Deliveries VALUES (@date, @emp, @deliverer, @total)";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlParameter dateParam = new SqlParameter("@date", date);

command.Parameters.Add(dateParam);

SqlParameter empParam = new SqlParameter("@emp", emp);

command.Parameters.Add(empParam);

SqlParameter delParam = new SqlParameter("@deliverer", deliverer);

command.Parameters.Add(delParam);

SqlParameter totalParam = new SqlParameter("@total", total);

command.Parameters.Add(totalParam);

int result = command.ExecuteNonQuery();

return result;

}

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

{

this.Close();

}

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

{

string date = dateTimePicker1.Text;

string employeeName = comboBox1.Text;

string[] employeeNameArray = employeeName.Split(new char[] { ' ' }, StringSplitOptions.RemoveEmptyEntries);

int employeeID = employee\_get\_ID(employeeNameArray);

string del\_name = comboBox2.Text;

int del\_id = deliverer\_get\_id(del\_name);

int total = int.Parse(textBox1.Text);

int result = addDelivery(date, employeeID, del\_id, total);

if (result == 1) MessageBox.Show("Success");

}

}

public partial class EmployeeCreationForm : Form

{

public EmployeeCreationForm()

{

InitializeComponent();

comboBox2Fill();

comboBox3Fill();

comboBox4Fill();

comboBox5Fill();

}

private void comboBox2Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT nationality\_name FROM Nationalaties";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox2.Items.Add(ds.Tables[0].Rows[i]["nationality\_name"]);

}

}

private void comboBox3Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT qualification\_name FROM Qualifications";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox3.Items.Add(ds.Tables[0].Rows[i]["qualification\_name"]);

}

}

private void comboBox4Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT position\_name FROM Positions";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox4.Items.Add(ds.Tables[0].Rows[i]["position\_name"]);

}

}

private void comboBox5Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT street\_name FROM Streets";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox5.Items.Add(ds.Tables[0].Rows[i]["street\_name"]);

}

}

private int sex\_get\_ID(string sex\_name)

{

int id = 0;

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select sex\_id, sex\_name FROM Sexes";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

object sex\_id = reader.GetValue(0);

object sexName = reader.GetValue(1);

if (sex\_name == sexName.ToString())

{

id = Convert.ToInt16(sex\_id);

}

}

return id;

}

private int nationality\_get\_ID(string nationalityName)

{

int id = 0;

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select nationality\_id, nationality\_name FROM Nationalaties";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

object nat\_id = reader.GetValue(0);

object nat\_name = reader.GetValue(1);

if (nationalityName == nat\_name.ToString())

{

id = Convert.ToInt16(nat\_id);

}

}

return id;

}

private int qualification\_get\_id(string qualName)

{

int id = 0;

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select qualification\_id, qualification\_name FROM Qualifications";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

object qual\_id = reader.GetValue(0);

object qual\_name = reader.GetValue(1);

if (qualName == qual\_name.ToString())

{

id = Convert.ToInt16(qual\_id);

}

}

return id;

}

private int position\_get\_id(string position\_name)

{

int id = 0;

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select position\_id, position\_name FROM Positions";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

object pos\_id = reader.GetValue(0);

object pos\_name = reader.GetValue(1);

if (position\_name == pos\_name.ToString())

{

id = Convert.ToInt16(pos\_id);

}

}

return id;

}

private int street\_get\_id(string strName)

{

int id = 0;

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select street\_id, street\_name FROM Streets";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

object str\_id = reader.GetValue(0);

object str\_name = reader.GetValue(1);

if (strName == str\_name.ToString())

{

id = Convert.ToInt16(str\_id);

}

}

return id;

}

private int addEmployee(string last\_name, string first\_name, string father\_name, string birthday, string hiring\_date, int sex\_id, int nat\_id, int qual\_id, string phone, int pos\_id, int street\_id, int house, int apt, int exp)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "INSERT INTO Employees VALUES (@last\_name, @first\_name, @father\_name, @birthday, @hiring\_date, @sex, @nat, @qual, @phone, @pos, @street, @house, @apt, @exp)";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlParameter lnameParam = new SqlParameter("@last\_name", last\_name);

command.Parameters.Add(lnameParam);

SqlParameter fnameParam = new SqlParameter("@first\_name", first\_name);

command.Parameters.Add(fnameParam);

SqlParameter fthnameParam = new SqlParameter("@father\_name", father\_name);

command.Parameters.Add(fthnameParam);

SqlParameter bdayParam = new SqlParameter("@birthday", birthday);

command.Parameters.Add(bdayParam);

SqlParameter hdayParam = new SqlParameter("@hiring\_date", hiring\_date);

command.Parameters.Add(hdayParam);

SqlParameter sexParam = new SqlParameter("@sex", sex\_id);

command.Parameters.Add(sexParam);

SqlParameter natParam = new SqlParameter("@nat", nat\_id);

command.Parameters.Add(natParam);

SqlParameter qualParam = new SqlParameter("@qual", qual\_id);

command.Parameters.Add(qualParam);

SqlParameter phoneParam = new SqlParameter("@phone", phone);

command.Parameters.Add(phoneParam);

SqlParameter posParam = new SqlParameter("@pos", pos\_id);

command.Parameters.Add(posParam);

SqlParameter strParam = new SqlParameter("@street", street\_id);

command.Parameters.Add(strParam);

SqlParameter houseParam = new SqlParameter("@house", house);

command.Parameters.Add(houseParam);

SqlParameter aptParam = new SqlParameter("@apt", apt);

command.Parameters.Add(aptParam);

SqlParameter expParam = new SqlParameter("@exp", exp);

command.Parameters.Add(expParam);

int result = command.ExecuteNonQuery();

return result;

}

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

{

this.Close();

}

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

{

string last\_name = textBox1.Text;

string first\_name = textBox2.Text;

string father\_name = textBox3.Text;

string sex\_name = comboBox1.Text;

int sex\_id = sex\_get\_ID(sex\_name);

string birthday = dateTimePicker1.Text;

string qualification\_name = comboBox3.Text;

int qualification\_id = qualification\_get\_id(qualification\_name);

string hiring\_date = dateTimePicker2.Text;

string position\_name = comboBox4.Text;

int position\_id = position\_get\_id(position\_name);

int experience = int.Parse(textBox7.Text);

string phone\_number = textBox4.Text;

string street\_name = comboBox5.Text;

int street\_id = street\_get\_id(street\_name);

int house\_number = int.Parse(textBox5.Text);

int apt\_number = int.Parse(textBox6.Text);

string nationality\_name = comboBox2.Text;

int nationality\_id = nationality\_get\_ID(nationality\_name);

int result = addEmployee(last\_name, first\_name, father\_name, birthday, hiring\_date, sex\_id, nationality\_id, qualification\_id, phone\_number, position\_id, street\_id, house\_number, apt\_number, experience);

if (result == 1) MessageBox.Show("Success");

}

}

public partial class EmployeePositionGenerateList : Form

{

public EmployeePositionGenerateList()

{

InitializeComponent();

fillDataGridView();

}

private void fillDataGridView()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true;";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

string sqlExpression = "Select last\_name, first\_name, position\_name FROM Employees INNER JOIN Positions On Positions.position\_id = Employees.position\_fk ORDER BY position\_id";

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

dataGridView1.DataSource = ds.Tables[0];

}

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

{

this.Close();

}

}

public partial class EmployeeViewForm : Form

{

public EmployeeViewForm()

{

InitializeComponent();

fillDataGridView();

comboBox2Fill();

comboBox3Fill();

}

private void comboBox2Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT nationality\_name FROM Nationalaties";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox2.Items.Add(ds.Tables[0].Rows[i]["nationality\_name"]);

}

}

private void comboBox3Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT qualification\_name FROM Qualifications";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox3.Items.Add(ds.Tables[0].Rows[i]["qualification\_name"]);

}

}

private void fillDataGridView()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true;";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

string sqlExpression = "Select employee\_id, last\_name, first\_name, father\_name, birthday\_date, hiring\_date, sex\_name, nationality\_name, qualification\_name, phone\_number, position\_name, street\_name, house\_number, apt\_number, experience From Employees Inner Join Sexes on Employees.sex\_fk = Sexes.sex\_id Inner Join Nationalaties On Employees.nationality\_fk = Nationalaties.nationality\_id Inner Join Qualifications On Employees.qualification\_fk = Qualifications.qualification\_id Inner Join Positions On Employees.position\_fk = Positions.position\_id Inner Join Streets On Employees.street\_fk = Streets.street\_id";

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

dataGridView1.DataSource = ds.Tables[0];

}

private void fillDataGridView1(string name)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true;";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

string sqlExpression = "Select employee\_id, last\_name, first\_name, father\_name, birthday\_date, hiring\_date, sex\_name, nationality\_name, qualification\_name, phone\_number, position\_name, street\_name, house\_number, apt\_number, experience From Employees Inner Join Sexes on Employees.sex\_fk = Sexes.sex\_id Inner Join Nationalaties On Employees.nationality\_fk = Nationalaties.nationality\_id Inner Join Qualifications On Employees.qualification\_fk = Qualifications.qualification\_id Inner Join Positions On Employees.position\_fk = Positions.position\_id Inner Join Streets On Employees.street\_fk = Streets.street\_id WHERE sex\_name = @name";

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataAdapter adapter = new SqlDataAdapter(command);

SqlParameter nameParam = new SqlParameter("@name", name);

adapter.SelectCommand.Parameters.Add(nameParam);

DataSet ds = new DataSet();

adapter.Fill(ds);

dataGridView1.DataSource = ds.Tables[0];

}

private void fillDataGridView2(string name)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true;";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

string sqlExpression = "Select employee\_id, last\_name, first\_name, father\_name, birthday\_date, hiring\_date, sex\_name, nationality\_name, qualification\_name, phone\_number, position\_name, street\_name, house\_number, apt\_number, experience From Employees Inner Join Sexes on Employees.sex\_fk = Sexes.sex\_id Inner Join Nationalaties On Employees.nationality\_fk = Nationalaties.nationality\_id Inner Join Qualifications On Employees.qualification\_fk = Qualifications.qualification\_id Inner Join Positions On Employees.position\_fk = Positions.position\_id Inner Join Streets On Employees.street\_fk = Streets.street\_id WHERE nationality\_name = @name";

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataAdapter adapter = new SqlDataAdapter(command);

SqlParameter nameParam = new SqlParameter("@name", name);

adapter.SelectCommand.Parameters.Add(nameParam);

DataSet ds = new DataSet();

adapter.Fill(ds);

dataGridView1.DataSource = ds.Tables[0];

}

private void fillDataGridView3(string name)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true;";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

string sqlExpression = "Select employee\_id, last\_name, first\_name, father\_name, birthday\_date, hiring\_date, sex\_name, nationality\_name, qualification\_name, phone\_number, position\_name, street\_name, house\_number, apt\_number, experience From Employees Inner Join Sexes on Employees.sex\_fk = Sexes.sex\_id Inner Join Nationalaties On Employees.nationality\_fk = Nationalaties.nationality\_id Inner Join Qualifications On Employees.qualification\_fk = Qualifications.qualification\_id Inner Join Positions On Employees.position\_fk = Positions.position\_id Inner Join Streets On Employees.street\_fk = Streets.street\_id WHERE qualification\_name = @name";

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataAdapter adapter = new SqlDataAdapter(command);

SqlParameter nameParam = new SqlParameter("@name", name);

adapter.SelectCommand.Parameters.Add(nameParam);

DataSet ds = new DataSet();

adapter.Fill(ds);

dataGridView1.DataSource = ds.Tables[0];

}

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

{

this.Close();

}

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

{

string name = comboBox1.Text;

fillDataGridView1(name);

}

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

{

string name = comboBox2.Text;

fillDataGridView2(name);

}

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

{

string name = comboBox3.Text;

fillDataGridView3(name);

}

}

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

string login, password;

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

{

this.Close();

}

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

{

aboutForm aF = new aboutForm();

aF.Show();

}

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

{

login = textBox1.Text;

password = textBox2.Text;

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT \* FROM Logins";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

object loginDB = reader.GetValue(1);

object passwordDB = reader.GetValue(2);

if (login == loginDB.ToString() && password == passwordDB.ToString())

{

MainMenuForm mMF = new MainMenuForm(login);

this.Hide();

mMF.Show();

break;

}

else if (login != loginDB.ToString() || password != passwordDB.ToString())

{

MessageBox.Show("Проверьте правильность вводимых данных");

}

}

}

}

public partial class GroupCreationForm : Form

{

public GroupCreationForm()

{

InitializeComponent();

comboBox1Fill();

comboBox2Fill();

}

private void comboBox1Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT ingridient\_name FROM Ingridients";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox1.Items.Add(ds.Tables[0].Rows[i]["ingridient\_name"]);

}

}

private void comboBox2Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select FORMAT(delivery\_date, 'yyyy-MM-dd') AS Del FROM Deliveries";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox2.Items.Add(ds.Tables[0].Rows[i]["Del"]);

}

}

private int ingridient\_get\_id(string ingrName)

{

int id = 0;

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select ingridient\_id, ingridient\_name FROM Ingridients";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

object ingr\_id = reader.GetValue(0);

object ingr\_name = reader.GetValue(1);

if (ingrName == ingr\_name.ToString())

{

id = Convert.ToInt16(ingr\_id);

}

}

return id;

}

private int delivery\_get\_id(string delDate)

{

int id = 0;

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select delivery\_id, FORMAT(delivery\_date, 'yyyy-MM-dd') FROM Deliveries";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

object del\_id = reader.GetValue(0);

object del\_date = reader.GetValue(1);

if (delDate == del\_date.ToString())

{

id = Convert.ToInt16(del\_id);

}

}

return id;

}

private int addGroup(int quan, int ingr, int del, int price)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "INSERT INTO Groups VALUES (@quan, @ingr, @del, @price)";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlParameter quanParam = new SqlParameter("@quan", quan);

command.Parameters.Add(quanParam);

SqlParameter ingrParam = new SqlParameter("@ingr", ingr);

command.Parameters.Add(ingrParam);

SqlParameter delParam = new SqlParameter("@del", del);

command.Parameters.Add(delParam);

SqlParameter priceParam = new SqlParameter("@price", price);

command.Parameters.Add(priceParam);

int result = command.ExecuteNonQuery();

return result;

}

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

{

this.Close();

}

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

{

int quantity = int.Parse(textBox1.Text);

string ingridient\_name = comboBox1.Text;

int ingridient\_id = ingridient\_get\_id(ingridient\_name);

string delivery\_date = comboBox2.Text;

int delivery\_id = delivery\_get\_id(delivery\_date);

int price = int.Parse(textBox2.Text);

int result = addGroup(quantity, ingridient\_id, delivery\_id, price);

if (result == 1) MessageBox.Show("Success");

}

}

public partial class GroupViewForm : Form

{

public GroupViewForm()

{

InitializeComponent();

fillDataGridView();

}

private void fillDataGridView()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true;";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

string sqlExpression = "Select group\_id, quantity, ingridient\_name, delivery\_date, price FROM Groups Inner Join Ingridients ON Groups.ingridient\_fk = Ingridients.ingridient\_id Inner Join Deliveries ON Groups.delivery\_fk = Deliveries.delivery\_id";

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

dataGridView1.DataSource = ds.Tables[0];

}

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

{

this.Close();

}

}

public partial class IngridientCreationForm : Form

{

public IngridientCreationForm()

{

InitializeComponent();

comboBox1Fill();

comboBox2Fill();

}

private void comboBox1Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT ingridient\_type\_name FROM Ingridient\_types";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox1.Items.Add(ds.Tables[0].Rows[i]["ingridient\_type\_name"]);

}

}

private void comboBox2Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT measurment\_type FROM Measurement";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox2.Items.Add(ds.Tables[0].Rows[i]["measurment\_type"]);

}

}

private int ingridientType\_get\_id(string ingType)

{

int id = 0;

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select ingridient\_type\_id, ingridient\_type\_name FROM Ingridient\_types";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

object ing\_id = reader.GetValue(0);

object ing\_type = reader.GetValue(1);

if (ingType == ing\_type.ToString())

{

id = Convert.ToInt16(ing\_id);

}

}

return id;

}

private int measurement\_get\_id(string measType)

{

int id = 0;

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select measurement\_id, measurment\_type FROM Measurement";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

object meas\_id = reader.GetValue(0);

object meas\_type = reader.GetValue(1);

if (measType == meas\_type.ToString())

{

id = Convert.ToInt16(meas\_id);

}

}

return id;

}

private int addIngridient(string name, int type, int meas)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "INSERT INTO Ingridients VALUES (@name, @type, @meas)";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlParameter nameParam = new SqlParameter("@name", name);

command.Parameters.Add(nameParam);

SqlParameter typeParam = new SqlParameter("@type", type);

command.Parameters.Add(typeParam);

SqlParameter measParam = new SqlParameter("@meas", meas);

command.Parameters.Add(measParam);

int result = command.ExecuteNonQuery();

return result;

}

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

{

this.Close();

}

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

{

string name = textBox1.Text;

string type = comboBox1.Text;

int ing\_type = ingridientType\_get\_id(type);

string meas = comboBox2.Text;

int meas\_type = measurement\_get\_id(meas);

int result = addIngridient(name, ing\_type, meas\_type);

if (result == 1) MessageBox.Show("Success");

}

}

public partial class IngridientsViewForm : Form

{

public IngridientsViewForm()

{

InitializeComponent();

fillDataGridView();

comboBox1Fill();

}

private void comboBox1Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT ingridient\_type\_name FROM Ingridient\_types";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox1.Items.Add(ds.Tables[0].Rows[i]["ingridient\_type\_name"]);

}

}

private void fillDataGridView()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true;";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

string sqlExpression = "Select ingridient\_id, ingridient\_name, ingridient\_type\_name, measurment\_type From Ingridients Inner Join Ingridient\_types On Ingridients.ingridient\_type\_fk = Ingridient\_types.ingridient\_type\_id Inner Join Measurement On Ingridients.measurement\_fk = Measurement.measurement\_id";

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

dataGridView1.DataSource = ds.Tables[0];

}

private void fillDataGridView1(string name)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true;";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

string sqlExpression = "Select ingridient\_id, ingridient\_name, ingridient\_type\_name, measurment\_type From Ingridients Inner Join Ingridient\_types On Ingridients.ingridient\_type\_fk = Ingridient\_types.ingridient\_type\_id Inner Join Measurement On Ingridients.measurement\_fk = Measurement.measurement\_id WHERE ingridient\_type\_name = @name";

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataAdapter adapter = new SqlDataAdapter(command);

SqlParameter nameParam = new SqlParameter("@name", name);

adapter.SelectCommand.Parameters.Add(nameParam);

DataSet ds = new DataSet();

adapter.Fill(ds);

dataGridView1.DataSource = ds.Tables[0];

}

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

{

this.Close();

}

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

{

string name = comboBox1.Text;

fillDataGridView1(name);

}

}

public partial class MainMenuForm : Form

{

string user;

object start = 0, end = 0;

public MainMenuForm(string username)

{

InitializeComponent();

this.user = username;

label2.Text += user;

}

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

{

Form1 frm = new Form1();

this.Close();

frm.Show();

}

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

{

aboutForm aF = new aboutForm();

aF.Show();

}

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

{

mealCardCreationForm mCCF = new mealCardCreationForm();

mCCF.Show();

}

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

{

mealCardViewForm mCVF = new mealCardViewForm();

mCVF.Show();

}

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

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select (last\_name + ' ' + first\_name) AS Employee, meal\_name, start\_datetime, finish\_datetime FROM Meal\_Preparations INNER JOIN Employees ON Employees.employee\_id = Meal\_Preparations.employee\_fk INNER JOIN Meals ON Meals.meal\_id = Meal\_Preparations.meal\_fk";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

Word.\_Application application;

Word.\_Document document;

Object missingObj = System.Reflection.Missing.Value;

Object trueObj = true;

Object falseObj = false;

application = new Word.Application();

Object templatePathObj = @"C:\123.docx";

try

{

document = application.Documents.Add(ref templatePathObj, ref missingObj, ref missingObj, ref missingObj);

}

catch (Exception error)

{

application.Quit(ref missingObj, ref missingObj, ref missingObj);

document = null;

application = null;

throw error;

}

application.Visible = true;

Word.Range rng = application.ActiveDocument.Range(0, 0);

rng.InsertBefore("Prepared meals");

rng.Font.Name = "Times New Roman";

rng.Font.Size = 14;

rng.InsertParagraphAfter();

rng.InsertParagraphAfter();

rng.SetRange(rng.End, rng.End);

rng.Tables.Add(application.ActiveDocument.Paragraphs[2].Range, 2, 4, ref missingObj, ref missingObj);

Word.Table tbl = application.ActiveDocument.Tables[1];

tbl.Borders.OutsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;

tbl.Borders.InsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;

tbl.Cell(1, 1).Range.Text = "id";

tbl.Cell(1, 2).Range.Text = "employer";

tbl.Cell(1, 3).Range.Text = "start";

tbl.Cell(1, 4).Range.Text = "finish";

int i = 2;

while (reader.Read())

{

object meal\_preparation\_id = reader.GetValue(0);

object employee\_fk = reader.GetValue(1);

object meal\_fk = reader.GetValue(2);

object start = reader.GetValue(3);

tbl.Cell(i, 1).Range.Text = meal\_preparation\_id.ToString();

tbl.Cell(i, 2).Range.Text = employee\_fk.ToString();

tbl.Cell(i, 3).Range.Text = meal\_fk.ToString();

tbl.Cell(i, 4).Range.Text = start.ToString();

i++;

tbl.Rows.Add(ref missingObj);

}

}

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

{

IngridientCreationForm iCF = new IngridientCreationForm();

iCF.Show();

}

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

{

IngridientsViewForm iVF = new IngridientsViewForm();

iVF.Show();

}

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

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select ingridient\_id, ingridient\_name, ingridient\_type\_name, measurment\_type From Ingridients Inner Join Ingridient\_types On Ingridients.ingridient\_type\_fk = Ingridient\_types.ingridient\_type\_id Inner Join Measurement On Ingridients.measurement\_fk = Measurement.measurement\_id";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

Word.\_Application application;

Word.\_Document document;

Object missingObj = System.Reflection.Missing.Value;

Object trueObj = true;

Object falseObj = false;

application = new Word.Application();

Object templatePathObj = @"C:\123.docx";

try

{

document = application.Documents.Add(ref templatePathObj, ref missingObj, ref missingObj, ref missingObj);

}

catch (Exception error)

{

application.Quit(ref missingObj, ref missingObj, ref missingObj);

document = null;

application = null;

throw error;

}

application.Visible = true;

Word.Range rng = application.ActiveDocument.Range(0, 0);

rng.InsertBefore("Ingridients");

rng.Font.Name = "Times New Roman";

rng.Font.Size = 14;

rng.InsertParagraphAfter();

rng.InsertParagraphAfter();

rng.SetRange(rng.End, rng.End);

rng.Tables.Add(application.ActiveDocument.Paragraphs[2].Range, 2, 4, ref missingObj, ref missingObj);

Word.Table tbl = application.ActiveDocument.Tables[1];

tbl.Borders.OutsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;

tbl.Borders.InsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;

tbl.Cell(1, 1).Range.Text = "id";

tbl.Cell(1, 2).Range.Text = "name";

tbl.Cell(1, 3).Range.Text = "type";

tbl.Cell(1, 4).Range.Text = "measurment";

int i = 2;

while (reader.Read())

{

object ingridient\_id = reader.GetValue(0);

object ingridient\_name = reader.GetValue(1);

object ingridient\_type = reader.GetValue(2);

object measurement = reader.GetValue(3);

tbl.Cell(i, 1).Range.Text = ingridient\_id.ToString();

tbl.Cell(i, 2).Range.Text = ingridient\_name.ToString();

tbl.Cell(i, 3).Range.Text = ingridient\_type.ToString();

tbl.Cell(i, 4).Range.Text = measurement.ToString();

i++;

tbl.Rows.Add(ref missingObj);

}

}

private void button7\_Click(object sender, EventArgs e)

{

MealCreationForm mCF = new MealCreationForm();

mCF.Show();

}

private void button8\_Click(object sender, EventArgs e)

{

MealViewForm mVF = new MealViewForm();

mVF.Show();

}

private void button9\_Click(object sender, EventArgs e)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select meal\_id, meal\_name, meal\_type\_name, cuisine\_name , mass, price, last\_name + ' ' + first\_name AS employee, recipe From Meals Inner Join Meal\_types On Meals.meal\_type\_fk = Meal\_types.meal\_type\_id Inner Join Cuisines On Meals.cuisine\_fk = Cuisines.cuisine\_id Inner Join Employees On Meals.employee\_fk = Employees.employee\_id";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

Word.\_Application application;

Word.\_Document document;

Object missingObj = System.Reflection.Missing.Value;

Object trueObj = true;

Object falseObj = false;

application = new Word.Application();

Object templatePathObj = @"C:\123.docx";

try

{

document = application.Documents.Add(ref templatePathObj, ref missingObj, ref missingObj, ref missingObj);

}

catch (Exception error)

{

application.Quit(ref missingObj, ref missingObj, ref missingObj);

document = null;

application = null;

throw error;

}

application.Visible = true;

Word.Range rng = application.ActiveDocument.Range(0, 0);

rng.InsertBefore("Meals");

rng.Font.Name = "Times New Roman";

rng.Font.Size = 14;

rng.InsertParagraphAfter();

rng.InsertParagraphAfter();

rng.SetRange(rng.End, rng.End);

rng.Tables.Add(application.ActiveDocument.Paragraphs[2].Range, 2, 8, ref missingObj, ref missingObj);

Word.Table tbl = application.ActiveDocument.Tables[1];

tbl.Borders.OutsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;

tbl.Borders.InsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;

tbl.Cell(1, 1).Range.Text = "id";

tbl.Cell(1, 2).Range.Text = "name";

tbl.Cell(1, 3).Range.Text = "type";

tbl.Cell(1, 4).Range.Text = "cuisine";

tbl.Cell(1, 5).Range.Text = "mass";

tbl.Cell(1, 6).Range.Text = "price";

tbl.Cell(1, 7).Range.Text = "employer";

tbl.Cell(1, 8).Range.Text = "recipe";

int i = 2;

while (reader.Read())

{

object meal\_id = reader.GetValue(0);

object meal\_name = reader.GetValue(1);

object meal\_type = reader.GetValue(2);

object meal\_cuisine = reader.GetValue(3);

object meal\_mass = reader.GetValue(4);

object meal\_price = reader.GetValue(5);

object meal\_emp = reader.GetValue(6);

object meal\_recipe = reader.GetValue(7);

tbl.Cell(i, 1).Range.Text = meal\_id.ToString();

tbl.Cell(i, 2).Range.Text = meal\_name.ToString();

tbl.Cell(i, 3).Range.Text = meal\_type.ToString();

tbl.Cell(i, 4).Range.Text = meal\_cuisine.ToString();

tbl.Cell(i, 5).Range.Text = meal\_mass.ToString();

tbl.Cell(i, 6).Range.Text = meal\_price.ToString();

tbl.Cell(i, 7).Range.Text = meal\_emp.ToString();

tbl.Cell(i, 8).Range.Text = meal\_recipe.ToString();

i++;

tbl.Rows.Add(ref missingObj);

}

}

private void button10\_Click(object sender, EventArgs e)

{

ComponentCreationForm cCF = new ComponentCreationForm();

cCF.Show();

}

private void button11\_Click(object sender, EventArgs e)

{

ComponentsViewForm cVF = new ComponentsViewForm();

cVF.Show();

}

private void button12\_Click(object sender, EventArgs e)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select component\_id, component\_name, meal\_name, component\_type\_name, description From Components Inner Join Meals On Components.meal\_fk = Meals.meal\_id Inner Join Component\_types On Components.component\_type\_fk = Component\_types.component\_type\_id";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

Word.\_Application application;

Word.\_Document document;

Object missingObj = System.Reflection.Missing.Value;

Object trueObj = true;

Object falseObj = false;

application = new Word.Application();

Object templatePathObj = @"C:\123.docx";

try

{

document = application.Documents.Add(ref templatePathObj, ref missingObj, ref missingObj, ref missingObj);

}

catch (Exception error)

{

application.Quit(ref missingObj, ref missingObj, ref missingObj);

document = null;

application = null;

throw error;

}

application.Visible = true;

Word.Range rng = application.ActiveDocument.Range(0, 0);

rng.InsertBefore("Components");

rng.Font.Name = "Times New Roman";

rng.Font.Size = 14;

rng.InsertParagraphAfter();

rng.InsertParagraphAfter();

rng.SetRange(rng.End, rng.End);

rng.Tables.Add(application.ActiveDocument.Paragraphs[2].Range, 2, 5, ref missingObj, ref missingObj);

Word.Table tbl = application.ActiveDocument.Tables[1];

tbl.Borders.OutsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;

tbl.Borders.InsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;

tbl.Cell(1, 1).Range.Text = "id";

tbl.Cell(1, 2).Range.Text = "name";

tbl.Cell(1, 3).Range.Text = "meal";

tbl.Cell(1, 4).Range.Text = "type";

tbl.Cell(1, 5).Range.Text = "description";

int i = 2;

while (reader.Read())

{

object meal\_id = reader.GetValue(0);

object meal\_name = reader.GetValue(1);

object meal\_type = reader.GetValue(2);

object meal\_cuisine = reader.GetValue(3);

object meal\_mass = reader.GetValue(4);

tbl.Cell(i, 1).Range.Text = meal\_id.ToString();

tbl.Cell(i, 2).Range.Text = meal\_name.ToString();

tbl.Cell(i, 3).Range.Text = meal\_type.ToString();

tbl.Cell(i, 4).Range.Text = meal\_cuisine.ToString();

tbl.Cell(i, 5).Range.Text = meal\_mass.ToString();

i++;

tbl.Rows.Add(ref missingObj);

}

}

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

{

this.Close();

Form1 frm = new Form1();

frm.Close();

}

private void button13\_Click(object sender, EventArgs e)

{

EmployeeCreationForm eCF = new EmployeeCreationForm();

eCF.Show();

}

private void button14\_Click(object sender, EventArgs e)

{

EmployeeViewForm eVF = new EmployeeViewForm();

eVF.Show();

}

private void button15\_Click(object sender, EventArgs e)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select employee\_id, last\_name, first\_name, father\_name, birthday\_date, hiring\_date, sex\_name, nationality\_name, qualification\_name, phone\_number, position\_name, street\_name, house\_number, apt\_number, experience From Employees Inner Join Sexes on Employees.sex\_fk = Sexes.sex\_id Inner Join Nationalaties On Employees.nationality\_fk = Nationalaties.nationality\_id Inner Join Qualifications On Employees.qualification\_fk = Qualifications.qualification\_id Inner Join Positions On Employees.position\_fk = Positions.position\_id Inner Join Streets On Employees.street\_fk = Streets.street\_id";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

Word.\_Application application;

Word.\_Document document;

Object missingObj = System.Reflection.Missing.Value;

Object trueObj = true;

Object falseObj = false;

application = new Word.Application();

Object templatePathObj = @"C:\123.docx";

try

{

document = application.Documents.Add(ref templatePathObj, ref missingObj, ref missingObj, ref missingObj);

}

catch (Exception error)

{

application.Quit(ref missingObj, ref missingObj, ref missingObj);

document = null;

application = null;

throw error;

}

application.Visible = true;

Word.Range rng = application.ActiveDocument.Range(0, 0);

rng.InsertBefore("Employees");

rng.Font.Name = "Times New Roman";

rng.Font.Size = 14;

rng.InsertParagraphAfter();

rng.InsertParagraphAfter();

rng.SetRange(rng.End, rng.End);

rng.Tables.Add(application.ActiveDocument.Paragraphs[2].Range, 2, 15, ref missingObj, ref missingObj);

Word.Table tbl = application.ActiveDocument.Tables[1];

tbl.Borders.OutsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;

tbl.Borders.InsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;

tbl.Cell(1, 1).Range.Text = "id";

tbl.Cell(1, 2).Range.Text = "last name";

tbl.Cell(1, 3).Range.Text = "first name";

tbl.Cell(1, 4).Range.Text = "father name";

tbl.Cell(1, 5).Range.Text = "birthday";

tbl.Cell(1, 6).Range.Text = "hireday";

tbl.Cell(1, 7).Range.Text = "sex";

tbl.Cell(1, 8).Range.Text = "nationality";

tbl.Cell(1, 9).Range.Text = "qualification";

tbl.Cell(1, 10).Range.Text = "phone";

tbl.Cell(1, 11).Range.Text = "position";

tbl.Cell(1, 12).Range.Text = "street";

tbl.Cell(1, 13).Range.Text = "house";

tbl.Cell(1, 14).Range.Text = "apt";

tbl.Cell(1, 15).Range.Text = "experience";

int i = 2;

while (reader.Read())

{

object id = reader.GetValue(0);

object lname = reader.GetValue(1);

object fname = reader.GetValue(2);

object fthname = reader.GetValue(3);

object birthday = reader.GetValue(4);

object hireday = reader.GetValue(5);

object sex = reader.GetValue(6);

object nat = reader.GetValue(7);

object qual = reader.GetValue(8);

object phone = reader.GetValue(9);

object pos = reader.GetValue(10);

object street = reader.GetValue(11);

object house = reader.GetValue(12);

object apt = reader.GetValue(13);

object exp = reader.GetValue(14);

tbl.Cell(i, 1).Range.Text = id.ToString();

tbl.Cell(i, 2).Range.Text = lname.ToString();

tbl.Cell(i, 3).Range.Text = fname.ToString();

tbl.Cell(i, 4).Range.Text = fthname.ToString();

tbl.Cell(i, 5).Range.Text = birthday.ToString();

tbl.Cell(i, 6).Range.Text = hireday.ToString();

tbl.Cell(i, 7).Range.Text = sex.ToString();

tbl.Cell(i, 8).Range.Text = nat.ToString();

tbl.Cell(i, 9).Range.Text = qual.ToString();

tbl.Cell(i, 10).Range.Text = phone.ToString();

tbl.Cell(i, 11).Range.Text = pos.ToString();

tbl.Cell(i, 12).Range.Text = street.ToString();

tbl.Cell(i, 13).Range.Text = house.ToString();

tbl.Cell(i, 14).Range.Text = apt.ToString();

tbl.Cell(i, 15).Range.Text = exp.ToString();

i++;

tbl.Rows.Add(ref missingObj);

}

}

private void button16\_Click(object sender, EventArgs e)

{

positionCreationForm pCF = new positionCreationForm();

pCF.Show();

}

private void button17\_Click(object sender, EventArgs e)

{

positionsViewForm pVF = new positionsViewForm();

pVF.Show();

}

private void button18\_Click(object sender, EventArgs e)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select \* FROM Positions";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

Word.\_Application application;

Word.\_Document document;

Object missingObj = System.Reflection.Missing.Value;

Object trueObj = true;

Object falseObj = false;

application = new Word.Application();

Object templatePathObj = @"C:\123.docx";

try

{

document = application.Documents.Add(ref templatePathObj, ref missingObj, ref missingObj, ref missingObj);

}

catch (Exception error)

{

application.Quit(ref missingObj, ref missingObj, ref missingObj);

document = null;

application = null;

throw error;

}

application.Visible = true;

Word.Range rng = application.ActiveDocument.Range(0, 0);

rng.InsertBefore("Positions");

rng.Font.Name = "Times New Roman";

rng.Font.Size = 14;

rng.InsertParagraphAfter();

rng.InsertParagraphAfter();

rng.SetRange(rng.End, rng.End);

rng.Tables.Add(application.ActiveDocument.Paragraphs[2].Range, 2, 2, ref missingObj, ref missingObj);

Word.Table tbl = application.ActiveDocument.Tables[1];

tbl.Borders.OutsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;

tbl.Borders.InsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;

tbl.Cell(1, 1).Range.Text = "id";

tbl.Cell(1, 2).Range.Text = "position";

int i = 2;

while (reader.Read())

{

object id = reader.GetValue(0);

object name = reader.GetValue(1);

tbl.Cell(i, 1).Range.Text = id.ToString();

tbl.Cell(i, 2).Range.Text = name.ToString();

i++;

tbl.Rows.Add(ref missingObj);

}

}

private void button19\_Click(object sender, EventArgs e)

{

EmployeePositionGenerateList ePGL = new EmployeePositionGenerateList();

ePGL.Show();

}

private void button20\_Click(object sender, EventArgs e)

{

RevisionCreationForm rCF = new RevisionCreationForm();

rCF.Show();

}

private void button21\_Click(object sender, EventArgs e)

{

RevisionsViewForm rVF = new RevisionsViewForm();

rVF.Show();

}

private void button22\_Click(object sender, EventArgs e)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select revision\_id, start\_datetime, finish\_datetime, (last\_name + ' ' + first\_name) AS Employee, revision\_type\_name FROM Revisions INNER JOIN Employees ON Revisions.employee\_fk = Employees.employee\_id INNER JOIN Revision\_type ON Revisions.revision\_type\_fk = Revision\_type.revision\_type\_id";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

Word.\_Application application;

Word.\_Document document;

Object missingObj = System.Reflection.Missing.Value;

Object trueObj = true;

Object falseObj = false;

application = new Word.Application();

Object templatePathObj = @"C:\123.docx";

try

{

document = application.Documents.Add(ref templatePathObj, ref missingObj, ref missingObj, ref missingObj);

}

catch (Exception error)

{

application.Quit(ref missingObj, ref missingObj, ref missingObj);

document = null;

application = null;

throw error;

}

application.Visible = true;

Word.Range rng = application.ActiveDocument.Range(0, 0);

rng.InsertBefore("Revisions");

rng.Font.Name = "Times New Roman";

rng.Font.Size = 14;

rng.InsertParagraphAfter();

rng.InsertParagraphAfter();

rng.SetRange(rng.End, rng.End);

rng.Tables.Add(application.ActiveDocument.Paragraphs[2].Range, 2, 5, ref missingObj, ref missingObj);

Word.Table tbl = application.ActiveDocument.Tables[1];

tbl.Borders.OutsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;

tbl.Borders.InsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;

tbl.Cell(1, 1).Range.Text = "id";

tbl.Cell(1, 2).Range.Text = "start";

tbl.Cell(1, 3).Range.Text = "finish";

tbl.Cell(1, 4).Range.Text = "employee";

tbl.Cell(1, 5).Range.Text = "type";

int i = 2;

while (reader.Read())

{

object rev\_id = reader.GetValue(0);

object rev\_start = reader.GetValue(1);

object rev\_finish = reader.GetValue(2);

object emp = reader.GetValue(3);

object type = reader.GetValue(4);

tbl.Cell(i, 1).Range.Text = rev\_id.ToString();

tbl.Cell(i, 2).Range.Text = rev\_start.ToString();

tbl.Cell(i, 3).Range.Text = rev\_finish.ToString();

tbl.Cell(i, 4).Range.Text = emp.ToString();

tbl.Cell(i, 5).Range.Text = type.ToString();

i++;

tbl.Rows.Add(ref missingObj);

}

}

private void button23\_Click(object sender, EventArgs e)

{

DeliveryCreationForm dCF = new DeliveryCreationForm();

dCF.Show();

}

private void button24\_Click(object sender, EventArgs e)

{

DeliveriesViewForm dVF = new DeliveriesViewForm();

dVF.Show();

}

private void button25\_Click(object sender, EventArgs e)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select delivery\_id, delivery\_date, last\_name + ' ' + first\_name AS Employee, deliverer\_name, total From Deliveries INNER JOIN Employees ON Deliveries.employee\_fk = Employees.employee\_id Inner Join Deliverers On Deliveries.deliverer\_fk = Deliverers.deliverer\_id";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

Word.\_Application application;

Word.\_Document document;

Object missingObj = System.Reflection.Missing.Value;

Object trueObj = true;

Object falseObj = false;

application = new Word.Application();

Object templatePathObj = @"C:\123.docx";

try

{

document = application.Documents.Add(ref templatePathObj, ref missingObj, ref missingObj, ref missingObj);

}

catch (Exception error)

{

application.Quit(ref missingObj, ref missingObj, ref missingObj);

document = null;

application = null;

throw error;

}

application.Visible = true;

Word.Range rng = application.ActiveDocument.Range(0, 0);

rng.InsertBefore("Deliveries");

rng.Font.Name = "Times New Roman";

rng.Font.Size = 14;

rng.InsertParagraphAfter();

rng.InsertParagraphAfter();

rng.SetRange(rng.End, rng.End);

rng.Tables.Add(application.ActiveDocument.Paragraphs[2].Range, 2, 5, ref missingObj, ref missingObj);

Word.Table tbl = application.ActiveDocument.Tables[1];

tbl.Borders.OutsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;

tbl.Borders.InsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;

tbl.Cell(1, 1).Range.Text = "id";

tbl.Cell(1, 2).Range.Text = "date";

tbl.Cell(1, 3).Range.Text = "employee";

tbl.Cell(1, 4).Range.Text = "deliverer";

tbl.Cell(1, 5).Range.Text = "total";

int i = 2;

while (reader.Read())

{

object rev\_id = reader.GetValue(0);

object rev\_start = reader.GetValue(1);

object rev\_finish = reader.GetValue(2);

object emp = reader.GetValue(3);

object type = reader.GetValue(4);

tbl.Cell(i, 1).Range.Text = rev\_id.ToString();

tbl.Cell(i, 2).Range.Text = rev\_start.ToString();

tbl.Cell(i, 3).Range.Text = rev\_finish.ToString();

tbl.Cell(i, 4).Range.Text = emp.ToString();

tbl.Cell(i, 5).Range.Text = type.ToString();

i++;

tbl.Rows.Add(ref missingObj);

}

}

private void button26\_Click(object sender, EventArgs e)

{

DeliverersCreationForm dCF = new DeliverersCreationForm();

dCF.Show();

}

private void button27\_Click(object sender, EventArgs e)

{

DeliverersViewForm dVF = new DeliverersViewForm();

dVF.Show();

}

private void button28\_Click(object sender, EventArgs e)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select deliverer\_id,deliverer\_name, phone\_number, street\_name, house\_number, office\_number From Deliverers Inner Join Streets On Deliverers.street\_fk = Streets.street\_id";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

Word.\_Application application;

Word.\_Document document;

Object missingObj = System.Reflection.Missing.Value;

Object trueObj = true;

Object falseObj = false;

application = new Word.Application();

Object templatePathObj = @"C:\123.docx";

try

{

document = application.Documents.Add(ref templatePathObj, ref missingObj, ref missingObj, ref missingObj);

}

catch (Exception error)

{

application.Quit(ref missingObj, ref missingObj, ref missingObj);

document = null;

application = null;

throw error;

}

application.Visible = true;

Word.Range rng = application.ActiveDocument.Range(0, 0);

rng.InsertBefore("Deliverers");

rng.Font.Name = "Times New Roman";

rng.Font.Size = 14;

rng.InsertParagraphAfter();

rng.InsertParagraphAfter();

rng.SetRange(rng.End, rng.End);

rng.Tables.Add(application.ActiveDocument.Paragraphs[2].Range, 2, 6, ref missingObj, ref missingObj);

Word.Table tbl = application.ActiveDocument.Tables[1];

tbl.Borders.OutsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;

tbl.Borders.InsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;

tbl.Cell(1, 1).Range.Text = "id";

tbl.Cell(1, 2).Range.Text = "name";

tbl.Cell(1, 3).Range.Text = "phone";

tbl.Cell(1, 4).Range.Text = "street";

tbl.Cell(1, 5).Range.Text = "house";

tbl.Cell(1, 6).Range.Text = "office";

int i = 2;

while (reader.Read())

{

object rev\_id = reader.GetValue(0);

object rev\_start = reader.GetValue(1);

object rev\_finish = reader.GetValue(2);

object emp = reader.GetValue(3);

object type = reader.GetValue(4);

object office = reader.GetValue(5);

tbl.Cell(i, 1).Range.Text = rev\_id.ToString();

tbl.Cell(i, 2).Range.Text = rev\_start.ToString();

tbl.Cell(i, 3).Range.Text = rev\_finish.ToString();

tbl.Cell(i, 4).Range.Text = emp.ToString();

tbl.Cell(i, 5).Range.Text = type.ToString();

tbl.Cell(i, 6).Range.Text = office.ToString();

i++;

tbl.Rows.Add(ref missingObj);

}

}

private void button29\_Click(object sender, EventArgs e)

{

GroupCreationForm gCF = new GroupCreationForm();

gCF.Show();

}

private void button30\_Click(object sender, EventArgs e)

{

GroupViewForm gVF = new GroupViewForm();

gVF.Show();

}

private void button31\_Click(object sender, EventArgs e)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select group\_id, quantity, ingridient\_name, delivery\_date, price FROM Groups Inner Join Ingridients ON Groups.ingridient\_fk = Ingridients.ingridient\_id Inner Join Deliveries ON Groups.delivery\_fk = Deliveries.delivery\_id";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

Word.\_Application application;

Word.\_Document document;

Object missingObj = System.Reflection.Missing.Value;

Object trueObj = true;

Object falseObj = false;

application = new Word.Application();

Object templatePathObj = @"C:\123.docx";

try

{

document = application.Documents.Add(ref templatePathObj, ref missingObj, ref missingObj, ref missingObj);

}

catch (Exception error)

{

application.Quit(ref missingObj, ref missingObj, ref missingObj);

document = null;

application = null;

throw error;

}

application.Visible = true;

Word.Range rng = application.ActiveDocument.Range(0, 0);

rng.InsertBefore("Groups");

rng.Font.Name = "Times New Roman";

rng.Font.Size = 14;

rng.InsertParagraphAfter();

rng.InsertParagraphAfter();

rng.SetRange(rng.End, rng.End);

rng.Tables.Add(application.ActiveDocument.Paragraphs[2].Range, 2, 5, ref missingObj, ref missingObj);

Word.Table tbl = application.ActiveDocument.Tables[1];

tbl.Borders.OutsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;

tbl.Borders.InsideLineStyle = Word.WdLineStyle.wdLineStyleSingle;

tbl.Cell(1, 1).Range.Text = "id";

tbl.Cell(1, 2).Range.Text = "quantity";

tbl.Cell(1, 3).Range.Text = "ingridient";

tbl.Cell(1, 4).Range.Text = "date";

tbl.Cell(1, 5).Range.Text = "price";

int i = 2;

while (reader.Read())

{

object rev\_id = reader.GetValue(0);

object rev\_start = reader.GetValue(1);

object rev\_finish = reader.GetValue(2);

object emp = reader.GetValue(3);

object type = reader.GetValue(4);

tbl.Cell(i, 1).Range.Text = rev\_id.ToString();

tbl.Cell(i, 2).Range.Text = rev\_start.ToString();

tbl.Cell(i, 3).Range.Text = rev\_finish.ToString();

tbl.Cell(i, 4).Range.Text = emp.ToString();

tbl.Cell(i, 5).Range.Text = type.ToString();

i++;

tbl.Rows.Add(ref missingObj);

}

}

}

public partial class mealCardCreationForm : Form

{

public mealCardCreationForm()

{

InitializeComponent();

comboBox1Fill();

comboBox2Fill();

}

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

{

string employeeName = comboBox1.Text;

string[] employeeNameArray = employeeName.Split(new char[] { ' ' }, StringSplitOptions.RemoveEmptyEntries);

int employeeID = employee\_get\_ID(employeeNameArray);

string mealName = comboBox2.Text;

int mealID = meal\_get\_ID(mealName);

string startDateTime = dateTimePicker1.Text;

string finishDateTime = dateTimePicker2.Text;

int result = addMealCard(employeeID, mealID, startDateTime, finishDateTime);

if (result == 1) MessageBox.Show("Success");

}

private void comboBox1Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT last\_name, first\_name FROM Employees";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox1.Items.Add(ds.Tables[0].Rows[i]["last\_name"] + " " + ds.Tables[0].Rows[i]["first\_name"]);

}

}

private void comboBox2Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT meal\_name FROM Meals";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox2.Items.Add(ds.Tables[0].Rows[i]["meal\_name"]);

}

}

private int employee\_get\_ID(string[] empName)

{

int id = 0;

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select employee\_id, last\_name, first\_name FROM Employees";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

object employeeID = reader.GetValue(0);

object employee\_last\_name = reader.GetValue(1);

object employee\_first\_name = reader.GetValue(2);

if (employee\_last\_name.ToString() == empName[0] && employee\_first\_name.ToString() == empName[1])

{

id = Convert.ToInt16(employeeID);

}

}

return id;

}

private int meal\_get\_ID(string meal\_name)

{

int id = 0;

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select meal\_id, meal\_name FROM Meals";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

object mealID = reader.GetValue(0);

object mealName = reader.GetValue(1);

if (meal\_name == mealName.ToString())

{

id = Convert.ToInt16(mealID);

}

}

return id;

}

private int addMealCard(int employeeID, int mealID, string start, string finish)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "INSERT INTO Meal\_Preparations VALUES (@employeeID, @mealID, @start, @finish)";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlParameter empParam = new SqlParameter("@employeeID", employeeID);

command.Parameters.Add(empParam);

SqlParameter mealParam = new SqlParameter("@mealID", mealID);

command.Parameters.Add(mealParam);

SqlParameter startParam = new SqlParameter("@start", start);

command.Parameters.Add(startParam);

SqlParameter finishParam = new SqlParameter("@finish", finish);

command.Parameters.Add(finishParam);

int result = command.ExecuteNonQuery();

return result;

}

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

{

this.Close();

}

}

public partial class mealCardViewForm : Form

{

public mealCardViewForm()

{

InitializeComponent();

fillDataGridView();

comboBox1Fill();

}

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

{

this.Close();

}

private void fillDataGridView()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true;";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

string sqlExpression = "Select meal\_preparation\_id, (last\_name + ' ' + first\_name) AS Employee, meal\_name, start\_datetime, finish\_datetime FROM Meal\_Preparations INNER JOIN Employees ON Employees.employee\_id = Meal\_Preparations.employee\_fk INNER JOIN Meals ON Meals.meal\_id = Meal\_Preparations.meal\_fk";

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

dataGridView1.DataSource = ds.Tables[0];

}

private void fillDataGridView1(string date)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true;";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

string sqlExpression = "Select meal\_preparation\_id, (last\_name + ' ' + first\_name) AS Employee, meal\_name, start\_datetime, finish\_datetime FROM Meal\_Preparations INNER JOIN Employees ON Employees.employee\_id = Meal\_Preparations.employee\_fk INNER JOIN Meals ON Meals.meal\_id = Meal\_Preparations.meal\_fk WHERE start\_datetime >= @date";

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataAdapter adapter = new SqlDataAdapter(command);

SqlParameter dateParam = new SqlParameter("@date", date);

adapter.SelectCommand.Parameters.Add(dateParam);

DataSet ds = new DataSet();

adapter.Fill(ds);

dataGridView1.DataSource = ds.Tables[0];

}

private void fillDataGridView2(int id)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true;";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

string sqlExpression = "Select meal\_preparation\_id, (last\_name + ' ' + first\_name) AS Employee, meal\_name, start\_datetime, finish\_datetime FROM Meal\_Preparations INNER JOIN Employees ON Employees.employee\_id = Meal\_Preparations.employee\_fk INNER JOIN Meals ON Meals.meal\_id = Meal\_Preparations.meal\_fk WHERE employee\_id = @id";

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataAdapter adapter = new SqlDataAdapter(command);

SqlParameter idParam = new SqlParameter("@id", id);

adapter.SelectCommand.Parameters.Add(idParam);

DataSet ds = new DataSet();

adapter.Fill(ds);

dataGridView1.DataSource = ds.Tables[0];

}

private int employee\_get\_ID(string[] empName)

{

int id = 0;

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select employee\_id, last\_name, first\_name FROM Employees";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

object employeeID = reader.GetValue(0);

object employee\_last\_name = reader.GetValue(1);

object employee\_first\_name = reader.GetValue(2);

if (employee\_last\_name.ToString() == empName[0] && employee\_first\_name.ToString() == empName[1])

{

id = Convert.ToInt16(employeeID);

}

}

return id;

}

private void comboBox1Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT last\_name, first\_name FROM Employees";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox1.Items.Add(ds.Tables[0].Rows[i]["last\_name"] + " " + ds.Tables[0].Rows[i]["first\_name"]);

}

}

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

{

string date = dateTimePicker1.Text;

fillDataGridView1(date);

}

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

{

string employeeName = comboBox1.Text;

string[] employeeNameArray = employeeName.Split(new char[] { ' ' }, StringSplitOptions.RemoveEmptyEntries);

int employeeID = employee\_get\_ID(employeeNameArray);

fillDataGridView2(employeeID);

}

}

public partial class MealCreationForm : Form

{

public MealCreationForm()

{

InitializeComponent();

comboBox1Fill();

comboBox2Fill();

comboBox3Fill();

}

private void comboBox1Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT meal\_type\_name FROM Meal\_types";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox1.Items.Add(ds.Tables[0].Rows[i]["meal\_type\_name"]);

}

}

private void comboBox2Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT cuisine\_name FROM Cuisines";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox2.Items.Add(ds.Tables[0].Rows[i]["cuisine\_name"]);

}

}

private void comboBox3Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT last\_name, first\_name FROM Employees";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox3.Items.Add(ds.Tables[0].Rows[i]["last\_name"] + " " + ds.Tables[0].Rows[i]["first\_name"]);

}

}

private int mealType\_get\_ID(string meal\_type\_name)

{

int id = 0;

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select meal\_type\_id, meal\_type\_name FROM Meal\_types";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

object mealID = reader.GetValue(0);

object mealName = reader.GetValue(1);

if (meal\_type\_name == mealName.ToString())

{

id = Convert.ToInt16(mealID);

}

}

return id;

}

private int cuisine\_get\_ID(string cuisine\_name)

{

int id = 0;

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select cuisine\_id, cuisine\_name FROM Cuisines";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

object mealID = reader.GetValue(0);

object mealName = reader.GetValue(1);

if (cuisine\_name == mealName.ToString())

{

id = Convert.ToInt16(mealID);

}

}

return id;

}

private int employee\_get\_ID(string[] empName)

{

int id = 0;

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select employee\_id, last\_name, first\_name FROM Employees";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

object employeeID = reader.GetValue(0);

object employee\_last\_name = reader.GetValue(1);

object employee\_first\_name = reader.GetValue(2);

if (employee\_last\_name.ToString() == empName[0] && employee\_first\_name.ToString() == empName[1])

{

id = Convert.ToInt16(employeeID);

}

}

return id;

}

private int addMeal(string name, int type, int cuisine, int mass, int price, int employee, string recipe)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "INSERT INTO Meals VALUES (@name, @type, @cuisine, @mass, @price, @employee, @recipe)";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlParameter nameParam = new SqlParameter("@name", name);

command.Parameters.Add(nameParam);

SqlParameter typeParam = new SqlParameter("@type", type);

command.Parameters.Add(typeParam);

SqlParameter cuisineParam = new SqlParameter("@cuisine", cuisine);

command.Parameters.Add(cuisineParam);

SqlParameter massParam = new SqlParameter("@mass", mass);

command.Parameters.Add(massParam);

SqlParameter priceParam = new SqlParameter("@price", price);

command.Parameters.Add(priceParam);

SqlParameter employeeParam = new SqlParameter("@employee", employee);

command.Parameters.Add(employeeParam);

SqlParameter recipeParam = new SqlParameter("@recipe", recipe);

command.Parameters.Add(recipeParam);

int result = command.ExecuteNonQuery();

return result;

}

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

{

this.Close();

}

private void button1\_Click\_1(object sender, EventArgs e)

{

string name = textBox1.Text;

string type = comboBox1.Text;

int type\_id = mealType\_get\_ID(type);

string cuisine = comboBox2.Text;

int cuisine\_id = cuisine\_get\_ID(cuisine);

int mass = int.Parse(textBox2.Text);

int price = int.Parse(textBox3.Text);

string employeeName = comboBox3.Text;

string[] employeeNameArray = employeeName.Split(new char[] { ' ' }, StringSplitOptions.RemoveEmptyEntries);

int employee\_id = employee\_get\_ID(employeeNameArray);

string recipe = richTextBox1.Text;

int result = addMeal(name, type\_id, cuisine\_id, mass, price, employee\_id, recipe);

if (result == 1) MessageBox.Show("Success");

}

}

public partial class MealViewForm : Form

{

public MealViewForm()

{

InitializeComponent();

fillDataGridView();

comboBox1Fill();

comboBox2Fill();

}

private void comboBox1Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT meal\_type\_name FROM Meal\_types";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox1.Items.Add(ds.Tables[0].Rows[i]["meal\_type\_name"]);

}

}

private void comboBox2Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT cuisine\_name FROM Cuisines";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox2.Items.Add(ds.Tables[0].Rows[i]["cuisine\_name"]);

}

}

private void fillDataGridView()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true;";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

string sqlExpression = "Select meal\_id, meal\_name, meal\_type\_name, cuisine\_name , mass, price, last\_name + ' ' + first\_name AS employee, recipe From Meals Inner Join Meal\_types On Meals.meal\_type\_fk = Meal\_types.meal\_type\_id Inner Join Cuisines On Meals.cuisine\_fk = Cuisines.cuisine\_id Inner Join Employees On Meals.employee\_fk = Employees.employee\_id";

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

dataGridView1.DataSource = ds.Tables[0];

}

private void fillDataGridView1(string name)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true;";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

string sqlExpression = "Select meal\_id, meal\_name, meal\_type\_name, cuisine\_name , mass, price, last\_name + ' ' + first\_name AS employee, recipe From Meals Inner Join Meal\_types On Meals.meal\_type\_fk = Meal\_types.meal\_type\_id Inner Join Cuisines On Meals.cuisine\_fk = Cuisines.cuisine\_id Inner Join Employees On Meals.employee\_fk = Employees.employee\_id WHERE meal\_type\_name = @name";

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataAdapter adapter = new SqlDataAdapter(command);

SqlParameter nameParam = new SqlParameter("@name", name);

adapter.SelectCommand.Parameters.Add(nameParam);

DataSet ds = new DataSet();

adapter.Fill(ds);

dataGridView1.DataSource = ds.Tables[0];

}

private void fillDataGridView2(string name)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true;";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

string sqlExpression = "Select meal\_id, meal\_name, meal\_type\_name, cuisine\_name , mass, price, last\_name + ' ' + first\_name AS employee, recipe From Meals Inner Join Meal\_types On Meals.meal\_type\_fk = Meal\_types.meal\_type\_id Inner Join Cuisines On Meals.cuisine\_fk = Cuisines.cuisine\_id Inner Join Employees On Meals.employee\_fk = Employees.employee\_id WHERE cuisine\_name = @name";

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataAdapter adapter = new SqlDataAdapter(command);

SqlParameter nameParam = new SqlParameter("@name", name);

adapter.SelectCommand.Parameters.Add(nameParam);

DataSet ds = new DataSet();

adapter.Fill(ds);

dataGridView1.DataSource = ds.Tables[0];

}

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

{

this.Close();

}

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

{

string type = comboBox1.Text;

fillDataGridView1(type);

}

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

{

string cuisine = comboBox2.Text;

fillDataGridView2(cuisine);

}

}

public partial class RevisionCreationForm : Form

{

public RevisionCreationForm()

{

InitializeComponent();

comboBox1Fill();

comboBox2Fill();

}

private void comboBox1Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT last\_name, first\_name FROM Employees";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox1.Items.Add(ds.Tables[0].Rows[i]["last\_name"] + " " + ds.Tables[0].Rows[i]["first\_name"]);

}

}

private void comboBox2Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT revision\_type\_name FROM Revision\_type";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox2.Items.Add(ds.Tables[0].Rows[i]["revision\_type\_name"]);

}

}

private int revision\_type\_get\_id(string revType\_name)

{

int id = 0;

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select revision\_type\_id, revision\_type\_name FROM Revision\_type";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

object rev\_type\_id = reader.GetValue(0);

object rev\_type\_name = reader.GetValue(1);

if (revType\_name == rev\_type\_name.ToString())

{

id = Convert.ToInt16(rev\_type\_id);

}

}

return id;

}

private int employee\_get\_ID(string[] empName)

{

int id = 0;

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select employee\_id, last\_name, first\_name FROM Employees";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

object employeeID = reader.GetValue(0);

object employee\_last\_name = reader.GetValue(1);

object employee\_first\_name = reader.GetValue(2);

if (employee\_last\_name.ToString() == empName[0] && employee\_first\_name.ToString() == empName[1])

{

id = Convert.ToInt16(employeeID);

}

}

return id;

}

private int addRevision(string start, string finish, int emp, int type, string comment)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "INSERT INTO Revisions VALUES (@start, @finish, @emp, @type, @comment)";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlParameter startParam = new SqlParameter("@start", start);

command.Parameters.Add(startParam);

SqlParameter finishParam = new SqlParameter("@finish", finish);

command.Parameters.Add(finishParam);

SqlParameter empParam = new SqlParameter("@emp", emp);

command.Parameters.Add(empParam);

SqlParameter typeParam = new SqlParameter("@type", type);

command.Parameters.Add(typeParam);

SqlParameter comParam = new SqlParameter("@comment", comment);

command.Parameters.Add(comParam);

int result = command.ExecuteNonQuery();

return result;

}

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

{

this.Close();

}

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

{

string start\_datetime = dateTimePicker1.Text;

string finish\_datetime = dateTimePicker2.Text;

string employeeName = comboBox1.Text;

string[] employeeNameArray = employeeName.Split(new char[] { ' ' }, StringSplitOptions.RemoveEmptyEntries);

int employeeID = employee\_get\_ID(employeeNameArray);

string revision\_type = comboBox2.Text;

int revision\_type\_id = revision\_type\_get\_id(revision\_type);

string comment = richTextBox1.Text;

int result = addRevision(start\_datetime, finish\_datetime, employeeID, revision\_type\_id, comment);

if (result == 1) MessageBox.Show("Success");

}

}

public partial class RevisionsViewForm : Form

{

public RevisionsViewForm()

{

InitializeComponent();

fillDataGridView();

comboBox2Fill();

comboBox1Fill();

}

private void comboBox1Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT revision\_type\_name FROM Revision\_type";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox1.Items.Add(ds.Tables[0].Rows[i]["revision\_type\_name"]);

}

}

private void comboBox2Fill()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "SELECT last\_name, first\_name FROM Employees";

SqlConnection connect = new SqlConnection(connectionString);

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

int length = ds.Tables[0].Rows.Count;

for (int i = 0; i < length; i++)

{

comboBox2.Items.Add(ds.Tables[0].Rows[i]["last\_name"] + " " + ds.Tables[0].Rows[i]["first\_name"]);

}

}

private int employee\_get\_ID(string[] empName)

{

int id = 0;

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true";

string sqlExpression = "Select employee\_id, last\_name, first\_name FROM Employees";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataReader reader = command.ExecuteReader();

while (reader.Read())

{

object employeeID = reader.GetValue(0);

object employee\_last\_name = reader.GetValue(1);

object employee\_first\_name = reader.GetValue(2);

if (employee\_last\_name.ToString() == empName[0] && employee\_first\_name.ToString() == empName[1])

{

id = Convert.ToInt16(employeeID);

}

}

return id;

}

private void fillDataGridView()

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true;";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

string sqlExpression = "Select revision\_id, start\_datetime, finish\_datetime, (last\_name + ' ' + first\_name) AS Employee, revision\_type\_name FROM Revisions INNER JOIN Employees ON Revisions.employee\_fk = Employees.employee\_id INNER JOIN Revision\_type ON Revisions.revision\_type\_fk = Revision\_type.revision\_type\_id";

SqlDataAdapter adapter = new SqlDataAdapter(sqlExpression, connect);

DataSet ds = new DataSet();

adapter.Fill(ds);

dataGridView1.DataSource = ds.Tables[0];

}

private void fillDataGridView2(string name)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true;";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

string sqlExpression = "Select revision\_id, start\_datetime, finish\_datetime, (last\_name + ' ' + first\_name) AS Employee, revision\_type\_name FROM Revisions INNER JOIN Employees ON Revisions.employee\_fk = Employees.employee\_id INNER JOIN Revision\_type ON Revisions.revision\_type\_fk = Revision\_type.revision\_type\_id WHERE revision\_type\_name = @name";

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataAdapter adapter = new SqlDataAdapter(command);

SqlParameter nameParam = new SqlParameter("@name", name);

adapter.SelectCommand.Parameters.Add(nameParam);

DataSet ds = new DataSet();

adapter.Fill(ds);

dataGridView1.DataSource = ds.Tables[0];

}

private void fillDataGridView1(int id)

{

string connectionString = @"Server = MAX; Database = KitchenerWorkstation; Trusted\_Connection = true;";

SqlConnection connect = new SqlConnection(connectionString);

connect.Open();

string sqlExpression = "Select revision\_id, start\_datetime, finish\_datetime, (last\_name + ' ' + first\_name) AS Employee, revision\_type\_name FROM Revisions INNER JOIN Employees ON Revisions.employee\_fk = Employees.employee\_id INNER JOIN Revision\_type ON Revisions.revision\_type\_fk = Revision\_type.revision\_type\_id WHERE employee\_id = @id";

SqlCommand command = new SqlCommand(sqlExpression, connect);

SqlDataAdapter adapter = new SqlDataAdapter(command);

SqlParameter idParam = new SqlParameter("@id", id);

adapter.SelectCommand.Parameters.Add(idParam);

DataSet ds = new DataSet();

adapter.Fill(ds);

dataGridView1.DataSource = ds.Tables[0];

}

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

{

this.Close();

}

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

{

string name = comboBox1.Text;

fillDataGridView2(name);

}

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

{

string employeeName = comboBox2.Text;

string[] employeeNameArray = employeeName.Split(new char[] { ' ' }, StringSplitOptions.RemoveEmptyEntries);

int employeeID = employee\_get\_ID(employeeNameArray);

fillDataGridView1(employeeID);

}

}

**Додаток Ж**

**Діаграма класів**

