МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ВІДОКРЕМЛЕНИЙ СТРУКТУРНИЙ ПІДРОЗДІЛ

«ЗАПОРІЗЬКИЙ ЕЛЕКТРОТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ

НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ «ЗАПОРІЗЬКА ПОЛІТЕХНІКА»

Циклова комісія спеціальності

121 Інженерія програмного забезпечення

спеціалізація «Розробка програмного забезпечення»

### ДОДАТОК ДЛЯ ЗАБЕЗПЕЧЕННЯ РОБОТИ КУР’ЄРСЬКОЇ СЛУЖБИ

### Пояснювальна записка до курсової роботи

121.44.03.01 ПЗ

#### 

Викладач Алла СМІРНОВА

Члени комісії Ян БАСОК

Олеся ТКАЧУК

Студент гр. РПЗ 21 1/9 Даніїл БОЙКО

2024

**РЕФЕРАТ**

Пояснювальна записка до курсової роботи містить 46 сторінок,   
12 рисунків, 6 таблиць, 2 додатки, 10 джерел.

Метою курсової роботи є розробка програмного продукту для забезпечення роботи з документами кур’єрської служби.

Програмне забезпечення розроблене у середовищі Microsoft Visual   
Studio 2010 за допомогою візуальної мови програмування C#.

Програма працює на ПК під керуванням операційних систем Microsoft Windows 7/10. Програма має зручний, максимально орієнтований на кінцевого користувача інтерфейс.

В загальному розділі розглянуті такі питання: опис предметної області, постанова завдання, обґрунтування вибору середовища розробки системи та середовища функціонування системи.

В спеціальному розділі розглянуті основні рішення щодо реалізації компонентів системи, методика роботи користувача з системою, яка включає інструкцію програмісту та інструкцію оператору.

КУР’ЄРСЬКА СЛУЖБА, ЗАМОВЛЕННЯ, КЛІЄНТИ, КІЛЬКІСТЬ ПОЗИЦІЙ, C#, СУБД ACCESS, РЕЛЯЦІЙНА МОДЕЛЬ, ОПЕРАЦІЙНА СИСТЕМА WINDOWS, ІНТЕРФЕЙС

**ЗМІСТ**

Вступ 4

1Загальний розділ 6

* 1. Опис предметної області 6
  2. Постанова завдання 8
  3. Обґрунтування вибору середовища розробки системи 9
  4. Обґрунтування вибору середовища функціонування системи 14

1. Спеціальний розділ 17
   1. Основні рішення щодо реалізації компонентів системи 17
   2. Використані моделі даних 18
   3. Методика роботи користувача з системою 22
      1. Керівництво програміста 22
      2. Керівництво оператора 24

Висновки 30

Перелік джерел посилання 31

Додаток А (обов’язковий) Текст програми 32

Додаток Б (обов’язковий) Приклади звітів 45

**ВСТУП**

Метою створення додатку є забезпечення автоматизації роботи з документами кур’єрської служби.

Для прискорення наукового і технічного прогресу в усіх сферах життєдіяльності людини водночас з іншими заходами передбачається широке впровадження інформаційних технологій як фундаменту подальшого розвитку людства. Висока ефективність комп’ютерів, постійне розширення області їх застосування, підвищення рівня програмного забезпечення зумовлює його широке застосування в усіх сферах людської діяльності. Перехід до автоматизації виробничих процесів підприємства вимагає не тільки бажання і засобів, але і значної підготовчої роботи, як організаційної, так і методичної. Для цього насамперед необхідно точно визначити задачі автоматизації.

За допомогою сучасної електронно-обчислювальної техніки здійснюється автоматизація розв’язання майже всіх задач, що виникають в процесі роботи організації чи підприємства.

Задача автоматизації — підвищення якості роботи як окремих робітників, так і підприємства в цілому. Комп'ютер - це тільки інструмент, що дозволяє максимально повно використовувати кваліфікацію фахівця і максимально спрощувати щоденну рутинну роботу.

Забезпечення робітників організації автоматизованими робочими місцями дозволяє підвищити продуктивність праці, скоротити їхню чисельність і при цьому підвищити швидкість обробки інформації та її достовірність, що необхідно для ефективного планування та управління. Оскільки автоматизація підвищує не тільки якість аналізу інформації, але і якість бізнесу підприємства.

Прискорення науково-технічного прогреса (НТП) підвищило вимоги до загальноосвітньої, технічної, економічної, технологічної підготовки робітників і службовців, підвищення кваліфікації, оцінки діяльності працівників, раціонального використання персоналу [1].

Особливо велике значення має наявність відповідної програми для організації пошуку інформації в базі даних, що має великі об’єми, вибірка інформації за різними критеріями і т.і.

Мета створення даної програми – забезпечення роботи з документами кур’єрської служби.

Програма розроблена за допомогою середовища програмування   
Visual Studio 2010 на мові програмування C#.

1. **ЗАГАЛЬНИЙ РОЗДІЛ**

**1.1 Опис предметної області**

Предметну область можна визначити як сферу людської діяльності, виділену і описану згідно зі встановленими критеріями.

Під терміном «модель» будемо розуміти будь-який математичний об’єкт, який описує властивості системи, що досліджується.

При цьому під моделлю предметної області розуміється деяка система, що імітує структуру або функціонування досліджуваної предметної області і відповідає основній вимозі - бути адекватної цій області.

Головна ідея автоматизованої системи полягає в тому, щоб перенести спосіб, яким здійснюється організація замовлень, разом з усією пов’язаною з нею паперовою роботою, в комп’ютер, де нею значно простіше керувати та піддавати аналізу.

Предметною областю, яка потребує автоматизації, являється система забезпечення роботи з документами кур’єрської служби.

Робота кур’єра нині дуже популярна. У часи пікової зайнятості люди користуються послугами кур’єрів для доставлення найрізноманітніших речей — від їжі до ділових паперів.

Кур’єр доставляє пакунки замовникам. Кур’єр може працювати на авто (іноді компанія надає автомобіль), мопеді, велосипеді чи навіть пішки. Він доставляє маленькі конверти та великі коробки. Він привозить піцу о другій ночі та встигає доставити важливі папери точно до часу ділової зустрічі. Робота кур’єра передбачає багато спілкування з різними людьми та дуже багато подорожей, навіть якщо ці подорожі обмежені одним містом чи навіть районом.

Працювати кур’єром можна у компанії чи на самого себе. Можна вибрати роботу в кафе чи ресторані, у службі перевезень або знаходити замовлення самостійно.

Навички, потрібні кур’єру:

* швидкість. І це навіть не про швидкість руху. А про швидкість реакції, швидкість знаходження найкоротших маршрутів, швидкість прийняття рішень;
* вміння спілкуватися з людьми. Щоб отримувати чайові та взагалі мати гарну репутацію серед клієнтів, потрібно бути привітним та справляти приємне враження. Як правило, служби доставки цікавляться думкою своїх замовників щодо наданого сервісу;
* активність. Робота кур’єра — це постійних рух [2].

 Кур'єр повинен знати:

* правила обліку, зберігання і доставки за призначенням кореспонденції, пакетів, посилок та інших предметів.. Місцевість і розташування об'єктів, що обслуговуються;
* транспортну розв'язку міста, види і нумерацію транспорту;
* основи ділового етикету, навички ведення ділових (в т.ч. телефонних) переговорів;
* правила і норми охорони праці, техніки безпеки і протипожежного захисту;
* правила внутрішнього трудового розпорядку;

Призначення на посаду кур'єра і звільнення з посади провадиться наказом директора підприємства відповідно до чинного законодавства.  
Кур'єр підпорядковується безпосередньо начальнику відділу логістики.  
В період тимчасової відсутності кур'єра обов'язки виконує особа, призначена у встановленому порядку. Дана особа набуває відповідних прав і несе відповідальність за виконання зазначених обов'язків.

Функціональні обов’язки:

* доставляти за призначенням ділові папери, пакети, листи, книги і т.п. відповідно до кур'єрським талоном і вказівками керівника, а також їх отримання і доставка від інших організацій;
* своєчасно і самостійно уточнювати місцезнаходження адресатів, розробляти маршрути руху, погоджувати дні і години прийому в організаціях-адресатів;
* своєчасно отримувати довіреності, необхідні для виконання доручень;
* виконувати доручення в зазначені терміни, доповідати про виконання або про причини, що перешкоджають виконанню, керівнику кур'єрської служби або уповноваженій особі;
* щодня звітувати керівнику по кур'єрським талонами і усним дорученням про виконану роботу;
* підтримувати позитивний моральний клімат в офісі. Виявляти взаємовиручку, відповідальність, довіру, підтримку, оптимізм;
* виконувати окремі службові доручення безпосереднього керівника;
* виконувати правила трудового розпорядку, прийняті в компанії [3].

База даних розташовується у файлі .mdb. Якщо файл відсутній у поточному каталозі, відкривається відповідне діалогове вікно, яке дозволяє знайти і завантажити необхідний файл.

Після завантаження файлу бази даних відбувається з’єднання його з програмою за допомогою провайдера Microsoft Jet OleDb4.0, а потім відкриття відповідних таблиць. З таблиці інформація одержується за допомогою SQL-запитів, використовується здебільшого конструкція Select. Наприклад: SELECT \* FROM Справочник. Цей запит вибирає всі записи з таблиці Справочник.

**1.2 Постанова завдання**

Мета створення даної програми – є автоматизація роботи з документами кур’єрської служби. Найголовнішою задачею є створення відповідної бази даних, яка відбиває інформацію про клієнтів, замовлення, кур’єрів.

Після вивчення предметної області і створення відповідної бази даних необхідно створити програмне забезпечення, яке має наступні функції:

* перегляд інформації про товари, клієнтів, кур’єрів;
* можливість додавання, редагування та видалення інформацію;
* виконання замовлення;
* формування чеку;
* пошук конкретної інформації;
* створення звітів.

До основних вимог проектованої системи відносяться:

* виведення на екран вихідних форм;
* інтерфейс повинен бути виконаний у вигляді стандартних вікон Windows;
* програма повинна надавати можливість переходу з однієї форми на іншу.

До основних вимог до надійності програми відносяться наступні:

* коректна інтерпретація і зберігання інформації;
* перевірка на коректність введеної інформації.

Для нормальної роботи програми необхідно мати персональний комп’ютер з такими характеристиками:

* процесор з тактовою частотою не менш 1,5 ГГц;
* оперативна пам'ять не менш 2 Гб;
* вільний дисковий простір 100 Мб та простір для зберігання даних в базі.

Окрім цього повинні бути у наявності монітор, клавіатура, та маніпулятор «миша».

Для виведення на друк інформації потрібен принтер. На всі пристрої, що використовуються в системі потрібні драйвери цих пристроїв.

Програма повинна нормально функціонувати під керуванням операційної системи Microsoft Windows 7/10.

Для перенесення програми на інший комп’ютер достатньо скопіювати папку, яка містить завантажувальний файл програми та базу даних. Додаткових дій з інсталяції програма не потребує.

**1.3 Обґрунтування вибору середовища розробки системи**

Розробка програмного забезпечення для роботи з базою даних виконана в візуальному середовищі програмування Visual Studio 2010.

Середовище MS Visual Studio 2010 – це лінійка продуктів компанії Майкрософт, що включають інтегроване середовище розробки програмного забезпечення і ряд інших інструментальних засобів. Дані продукти дозволяють розробляти як консольні додатки, так і додатки з графічним інтерфейсом, в тому числі з підтримкою технології Windows Forms, а також Web-сайти, Web-додатки, Web-служби як в рідному, так і в керованому кодах для всіх платформ, підтримуваних Microsoft Windows, Windows Mobile, Windows CE, .NET Framework, .NET Compact Framework і Microsoft Silverlight.

Visual Studio 2010 – побачила світ 12 квітня 2010 разом з .NET Framework 4.0. Visual Studio включає підтримку мов C# 4.0 і Visual Basic. NET 10.0, а також мову F#, який була відсутня в попередніх версіях.

Під платформою Microsoft.NET слід розуміти інтегровану систему (інфраструктуру) засобів розробки, розгортання і виконання складних (як правило, розподілених) програмних систем.

Основа .Net – це Microsoft .Net Framework – своєрідний каркас, набір засобів і технологій для розробки і виконання програмних систем.

Базовий рівень підсистеми MS.NET Framework складає загальномовне середовище виконання (Common Language Runtime або CLR).

Середовище виконання CLR реалізує управління пам'яттю, типами даних, міжмовною взаємодією, розгортанням (deployment) застосувань.

Середовище CLR активізує виконуваний код, виконує для нього перевірку безпеки, розташовує цей код в пам'яті і виконує його. Важливою частиною роботи середовища CLR є управління пам'яттю, а саме автоматичне вивільнення пам'яті, що звільняється при роботі програм (збірка сміття).

В ході компіляції програма на .NET-сумісній мові програмування трансформується відповідно до наперед заданої узагальненої специфікації типів Common Type System (CTS). Система типів CTS повністю описує всі типи даних, які підтримуються середовищем виконання, визначає їх взаємозв'язки і зберігає їх відображення в систему типів .NET.

Програмні файли на мові CIL, одержувані після компіляції програм на алгоритмічних мовах платформи MS.Net, називаються збірками (assembly), інше їх найменування - переносимі виконувані файли (Portable Executable або PE).

Збірки є файлами з розширеннями exe або dll і складаються з безпосередньо програмного коду на мові CIL і додаткових службових даних, які називаються метаданими (у складі метаданих необхідна інформація про збірку – відомості про типи, дані про версію, посилання на зовнішні збірки і т.п.)

Збірки перед своїм виконанням повинні пройти певні налаштування для роботи в умовах конкретної вибраної платформи – для виконання таких налаштувань у складі середовища CLR є ряд JIT-компіляторів (Just-Іn-Time compilers), що викликаються для перекладу програмного коду на проміжній мові (CIL-кода) в машинний (native) код платформи виконання [5].

Мовою програмування була обрана мова С#. C# – це строго типізована об'єктно-орієнтована мова, покликана забезпечити оптимальне поєднання зручності, простоти, виразності і продуктивності.

C# - це мова програмування створена спеціально для роботи у середовищі Microsoft .NET Framework.

Далі – про мову C#....

Об'єкти БД в засновані на SQL [9]. Доступ до бази даних виконується за допомогою інтерфейсу ADO.

ADO (від англ. ActiveX Data Objects - "об'єкти даних ActiveX") - інтерфейс програмування додатків для доступу до даних, розроблений компанією Microsoft (MS Access, MS SQL Server) і заснований на технології компонентів ActiveX.

Іншими словами, це інтерфейс високого рівня для роботи з OLE DB (Object Linking and Embedding, DataBase), орієнтований на використання в прикладних програмах.

ADO дозволяє представляти дані з різноманітних джерел (реляційних баз даних, текстових файлів і т. і.) в об'єктно-орієнтованому виді.

Технологія ADO незалежна від конкретного сервера БД , має підтримку як локальних баз даних різних типів, так і деяких клієнт- серверних БД. Плюсів у цієї технології багато. Драйвери , розроблені корпорацією Microsoft для власних потреб, більш надійні, ніж драйвери сторонніх виробників. Тому якщо потрібно працювати з базами даних MS Access або для архітектури клієнт-сервер використовувати MS SQL Server, то використання ADO буде найкращим. Крім того, є плюс і в питанні розповсюдження програм - у всіх сучасних Windows вбудовані драйвери ADO. Іншими словами, програма буде працювати на будь-якому ПК, де встановлений Windows [6].

Для зберігання вхідної інформації використовується реляційна база даних MS Access.

Бази даних MS Access мають багато плюсів, часто програмісти воліють використовувати саме їх. По-перше, база даних MS Access - це один файл. Скільки б таблиць та індексів вона не містила, все це зберігається в одному єдиному файлі. А значить, таку базу даних легше обслуговувати - переносити на нове місце, робити резервні копії і так далі. Ще один плюс - імена полів в такій БД можна давати кирилицею [7] .

Таким чином, система візуального програмування C# є найбільш оптимальним інструментом для розробки даного програмного продукту.

**1.4 Обґрунтування вибору середовища функціонування системи**

Середовищем функціонування програмного продукту обрана операційна система Wіndоws 7.

Перевага надана саме цій операційній системі серед інших тому, що вона:

* надає багатий інструментарій для розробки додатків;
* багатозадачність системи дозволяє запускати додатки, що потребують багато часу, у фоновому режимі, а це підвищує ефективність використання ПК в цілому;
* робота у захищеному режимі дає програмісту можливість використовувати всю фізичну пам'ять EОМ, у ОС Wіndows (на відміну від однозадачних ОС) механізм захисту прозорий для програміста;
* є, в цілому, широко розповсюдженою системою.

Крім цього, Windows 7 має ще деякі переваги, на відміну, наприклад, від Windows 2000 та попередніх версій:

* вдосконалена система встановлення програмного забезпечення Windows Installer, система захисту Windows File Protection;
* висока масштабованість;
* поліпшений інтерфейс користувача;
* вдосконалені мережеві засоби;
* загальний базовий код для інтернаціональних версій;
* стабільність;
* продуктивність;
* вбудовані засоби захисту;
* широка підтримка апаратного забезпечення: Plug-and-Play, USB, FireWire;
* вдосконалена файлова система NTFS (шифрування, індексування);
* вбудований дефрагментатор;
* підтримка декількох моніторів.

Ще однією перевагою Windows 7 є висока продуктивність. Продуктивність ОС визначається не тільки ефективністю реалізації підсистем управління пам'яттю, кешування жорсткого диска, якістю драйверів пристроїв, але також і її потребами в ресурсах.

Windows представляє програмістам всі необхідні засоби для створення користувальницького інтерфейсу. В результаті чого інтерфейс Windows - програм в значному уніфікований [8].

Програмування в операційній системі має деякі особливості.

Операційні системи Windows дозволяють використовувати віртуальну пам’ять. Інтерфейс систем використовує багатий набір системних функцій в якості засобу доступу до ресурсів операційної системи. Ці функції мають назву інтерфейс прикладного програмування (Application Programming Interface – API) і викликаються програмами для організації взаємодії їз системою (виділення пам’яті, виведення даних на екран, створення вікон і т. ін.). Завдяки 32-бітній адресації та іншим перевагам над операційними системами попередніх версій кожний процес систем виконується в своєму власному просторі, ізольованому від інших процесів. Це означає, що помилки в одному процесі не можуть вплинути на виконання інших додатків [8].

Саме ці переваги дозволили обрати операційну систему Windows 7 у якості середовища функціонування розробленого додатку.

Таким чином, операційна система Windows 7 – це досить потужна, безпечна і стабільна система з великою кількістю переваг і великим потенціалом розвитку, може повністю забезпечити розробку програми, що відповідає всім вимогам, а також забезпечити нормальну роботу зі створеною програмою.

2. **СПЕЦІАЛЬНИЙ РОЗДІЛ**

**2.1 Основні рішення щодо реалізації компонентів системи**

При запуску програми завантажується головний модуль програми Main. Тут перевіряється наявність бази даних в поточному каталозі, пошук бази даних та з’єднання її з програмою за допомогою механізму ADO.

Головний модуль завантажує модулі роботи з таблицями бази даних.

Інтерфейс – це апаратно-програмні засоби, що забезпечують графічне відображення і обмін інформацією між людиною і комп'ютером.

Користувальницький інтерфейс (user interface, UI) є своєрідним комунікаційним каналом, по якому здійснюється взаємодія користувача і комп'ютера.

Кращий користувальницький інтерфейс – це такий інтерфейс, якому користувач не повинен приділяти багато уваги, майже не помічати його. Користувач просто працює, замість того, щоб розмірковувати, яку кнопку натиснути або де клацнути мишею. Такий інтерфейс називають прозорим – користувач ніби дивиться крізь нього на свою роботу.

Щоб створити ефективний інтерфейс, який робив би роботу з програмою приємною, треба розуміти, які завдання будуть вирішувати користувачі з допомогою даної програми і які вимоги до інтерфейсу можуть виникнути у користувачів [9].

При створення додатку оформленню інтерфейсу приділялась особлива увага, адже, значною мірою, саме він визначає успіх даної програми та попит на неї. При розробці зовнішнього виду додатку враховувалися основні вимоги до нього, такі як:

- інтерфейс повинен бути зручним та зрозумілим для будь-якого користу-вача;

- інтерфейс повинен бути оформлений у сприятливих та «м’яких» кольорах, для зменшення навантаження та напруженості зорової системи;

- інтерфейс повинен містити достатню кількість інформації для користувача про процес та функції, які автоматизуються, але водночас, він не повинен бути загромаджений великою кількістю даних;

- при розробці інтерфейсу потрібно враховувати стандарти, які характерні для оформлення Windows-подібного інтерфейсу.

За основу інтерфейсу програми було прийнято стандартний стиль оформлення: побудовано основну форму. Цей стиль відповідає принципу побудови інтерфейсу Windows та більшості додатків, а тому є найбільш розповсюдженим, а отже і зручним для користувача.

При побудові інтерфейсу розробленого програмного продукту було використано безліч компонентів середовища розробки C#. Дані компоненти приведено в таблиці 2.1.

Таблиця 2.1– Перелік використовуваних компонентів C#

|  |  |
| --- | --- |
| Назва компоненту | Призначення компоненту |
| DataGridView | Елемент управління надає персоналізовану таблицю для відображення даних |
| Button | Компонент, який забезпечує функції кнопки, що широко  застосовується в діалогових вікнах. Процедура обробки  задається в події Click |
| ComboBox | Являє елемент керування поле зі списком Windows. Властивість Items – колекція елементів списку |
| GroupBox | Являє собою елемент керування для Windows, який відображає рамку навколо групи елементів управління і, необов'язково, заголовок над нею |
| DateTimePicker | Являє елемент керування Windows, який дозволяє обрати дату і час і відобразити їх у вказаному форматі.  Зберігає значення у властивості Value |
| TextBox | Призначений для введення і виведення текстової інформації |
| Label | Напис на формі |
| RepotViewer | Компонент, призначений для створення звіту |

**2.2 Використані моделі даних**

Для розробки даного програмного продукту використано дві моделі даних: об'єктно-орієнтована та реляційна. Реляційну модель було побудовано шляхом перетворення об'єктно-орієнтованої моделі бази даних.

Об'єктно-орієнтована модель – модель, що реалізує об'єктно-орієнтований підхід. Вона обробляє дані як абстрактні об'єкти, наділені властивостями, у вигляді неструктурованих даних, і використовуючи методи взаємодії з іншими об'єктами навколишнього світу.

При проектуванні бази даних було виявлено 5 сутностей: «Кур’єри», «Клієнти», «Товар», «Кількість позицій», «Замовлення». Дані сутності пов’язані між собою за ключовими полями типом зв’язку один до багатьох.

Сутність «Кур’єри» призначена для зберігання інформації про робітників. В реалізації бази даних сутність представлено таблицею «Kur» з наступними атрибутами (таблиця 2.2).

Таблиця 2.2 – Структура таблиці «Kur»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назва поля | Тип поля | Розмір поля | Ключ | Опис |
| ID\_kur | Текстовий | 255 | Так | Призначене для уніка-льності записів у таб-лиці та містить код людини |
| FIO | Текстовий | 255 | - | Призначене для збере-ження інформації про ім’я працівника |
| Data\_ro | Текстовий | 255 | - | Призначене для збереження інформації про дату народження |
| Pasport | Текстовий | 255 | - | Призначене для збереження інформації про паспортні дані |
| Data\_work | Текстовий | 255 | - | Призначене для збереження інформації про дату прийняття на роботу |
| Phone | Текстовий | 255 | - | Призначене для збереження даних про телефон |

Сутність «Клієнти» призначена для зберігання інформації про клієнтів. В реалізації бази даних сутність представлено таблицею «Klient» з наступними атрибутами (таблиця 2.3).

Таблиця 2.3 – Структура таблиці «Klient»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назва поля | Тип поля | Розмір поля | Ключ | Опис |
| ID\_klient | Текстовий | 255 | - | Містить код людини |
| Name\_K | Текстовий | 255 | - | Призначене для збереження інформації про ім’я людини |
| Adress | Текстовий | 255 | - | Призначене для збереження інформації про адресу клієнта |
| Phone | Текстовий | 255 | - | Призначене для збереження інформації про телефон |

Опис полів таблиці «Товари» представлена в таблиці 2.4.

Таблиця 2.4 – Структура таблиці «Tovar»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назва поля | Тип поля | Розмір поля | Ключ | Опис |
| ID\_tovar | Текстовий | 255 | - | Містить код продукту |
| Nazva | Текстовий | 255 | - | Назва продукту |
| Cena | Текстовий | 255 | - | Цена продукту |
| Opis | Текстовий | 255 | - | Опис продуку |

Опис полів таблиці «Кількість позицій» приведена в таблиці 2.5.

Таблиця 2.5 – Структура таблиці «Kol\_position»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назва поля | Тип поля | Розмір поля | Ключ | Опис |
| ID\_tovar | Текстовий | 255 | - | Містить код продукту |
| ID\_zakaz | Текстовий | 255 | - | Містить код замовлення |
| Kol | Цифровий | Ціле | - | Містить кількість продукту |

Опис полів таблиці «Замовлення» приведена в таблиці 2.6.

Таблиця 2.6 – Структура таблиці «Zakaz»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назва поля | Тип поля | Розмір поля | Ключ | Опис |
| ID\_zakaz | Текстовий | 255 | - | Містить код замовлення |
| ID\_kur | Текстовий | 255 | - | Містить код кур’єра |
| ID\_klient | Текстовий | 255 | - | Містить код клієнта |
| Data\_dost | Текстовий | 255 | - | Дата доставки |
| Oplata | Текстовий | 255 | - | Оплата |

Реляційна модель представляє дані у вигляді простих таблиць, розбитих на рядки і стовпці, на перетині яких розташовані дані. Запити до таких таблиць повертають таблиці, які самі можуть ставати предметом подальших запитів. Кожна база даних може включати декілька таблиць [11].

Реляційна модель даних – це модель, яка представляє дані у вигляді таблиць між якими існують зв’язки.

Між таблицям реляційної моделі існують відносини.

Відношення можуть бути таких типів:

* один до одного – це тип зв’язку, при якому кожному екземпляру однієї сутності відповідає не більше одного екземпляра іншої;
* один до багатьох – це тип зв’язку, при якому кожному екземпляру однієї сутності відповідають декілька екземплярів іншої;
* багато до багатьох – це тип зв’язку, при якому декільком екземплярам однієї сутності відповідають декілька екземплярів іншої.

Процес перетворення об'єктно-орієнтованої моделі бази даних в реляційну модель включає створення реляційної таблиці для кожної моделі об'єктної множини. Окрема об’єктна множина стає самостійною реляційною таблицею, а атрибути кожної об'єктної множини стають полями реляційної таблиці бази даних [5].

Структуру реляційної моделі представлено на рисунку 2.1.

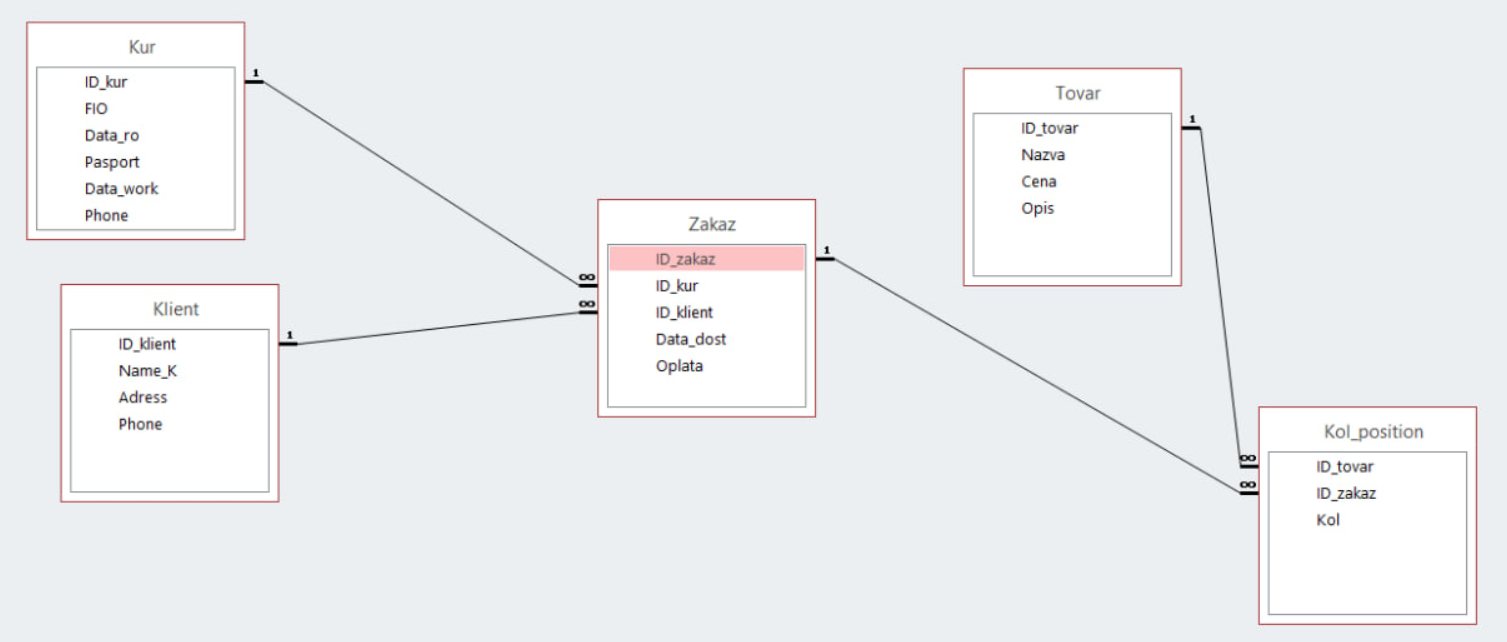


Рисунок 2.1 – Реляційна модель даних

**2.3 Методика роботи користувача з системою**

**2.3.1 Керівництво програміста**

Програма призначена для для забезпечення роботи з документами кур’єрської служби.

Програма створена в середовищі програмування Visual Studio 2010 на мові програмування C#.

База даних створена в СКБД Access.

Підключення таблиць бази даних на форми проекту виконано за допомогою Microsoft Jet 4.0 OLE DB Provider.

Контейнером даних являються компоненти ADO.

Для створення запитів використовувався вбудований відладчик SQL. Всі запити прописані програмно. Умовою надійної експлуатації програми є наявність бази даних в папці з відповідним .exe-файлом.

Для нормальної роботи програми необхідно мати персональний комп’ютер з такими характеристиками:

* процесор з тактовою частотою не менш 1,5 ГГц;
* оперативна пам'ять не менш 2 Гб;
* вільний дисковий простір 100 Мб та простір для зберігання даних в базі;

Окрім цього повинні бути у наявності монітор, клавіатура, та маніпулятор «миша».

Для виведення на друк інформації потрібен принтер. На всі пристрої, що використовуються в системі потрібні драйвери цих пристроїв.

Програма повинна нормально функціонувати під керуванням операційної системи Microsoft Windows 7/10.

Програмне забезпечення, яке повинно бути встановлено на персональному комп’ютері: операційна система Windows, CrystalReport.

Програма реалізує наступні функції:

* перегляд інформації про товари, клієнтів, кур’єрів;
* можливість додавання, редагування та видалення інформацію;
* виконання замовлення;
* формування чеку;
* пошук конкретної інформації;
* створення звітів.

Файл проекту має назву програми має назву courier.sln. База даних має назву courier.mdb.

Після кожної корекції бази даних за допомогою програми, необхідно робити архівну копію бази.

Після корегування модулів проекту необхідно перекомпілювати програму.

В програмі використовується подійний метод передачі керування та даних.

Вхідними даними до програми є файл бази даних courier.mdb.

Вихідними даними програми являється візуальні екранні форми та звіти.

Приклади звітів наведені в додатку Б.

**2.3.2 Керівництво оператора**

Програма призначена для забезпечення роботи з документами кур’єрської служби.

Програма створена в середовищі програмування Visual Studio 2010 на мові програмування C#.

База даних створена в СКБД Access.

Підключення таблиць бази даних на форми проекту виконано за допомогою Microsoft Jet 4.0 OLE DB Provider.

Контейнером даних являються компоненти ADO.

Для створення запитів використовувався вбудований відладчик SQL. Всі запити прописані програмно.

Умовою надійної експлуатації програми є наявність бази даних в папці з відповідним .exe-файлом.

Для нормальної роботи програми необхідно мати персональний комп’ютер з такими характеристиками:

* процесор з тактовою частотою не менш 1,5 ГГц;
* оперативна пам'ять не менш 2 Гб;
* вільний дисковий простір 100 Мб та простір для зберігання даних в базі;

Окрім цього повинні бути у наявності монітор, клавіатура, та маніпулятор «миша».

Для виведення на друк інформації потрібен принтер. На всі пристрої, що використовуються в системі потрібні драйвери цих пристроїв.

Програма повинна нормально функціонувати під керуванням операційної системи Microsoft Windows 7/10.

Програмне забезпечення, яке повинно бути встановлено на персональному комп’ютері: операційна система Windows, CrystalReport.

Для встановлення програмного продукту необхідно скопіювати файли автоматизованої системи на комп’ютер користувача в окрему папку.

Запуск програми відбувається із середовища Microsoft Windows шляхом відкриття файлу courier.exe.

Головна форма програми представлена на рисунку 2.2.



Рисунок 2.2 – Головна форма програми

При натисненні на кнопку «Товари» відкривається форма, представлена на рисунку 2.3.



Рисунок 2.3 – Форма для роботи з таблицею Товари

Тут можна додавати, редагувати, видаляти та шукати інформацію (рисунок 2.4).

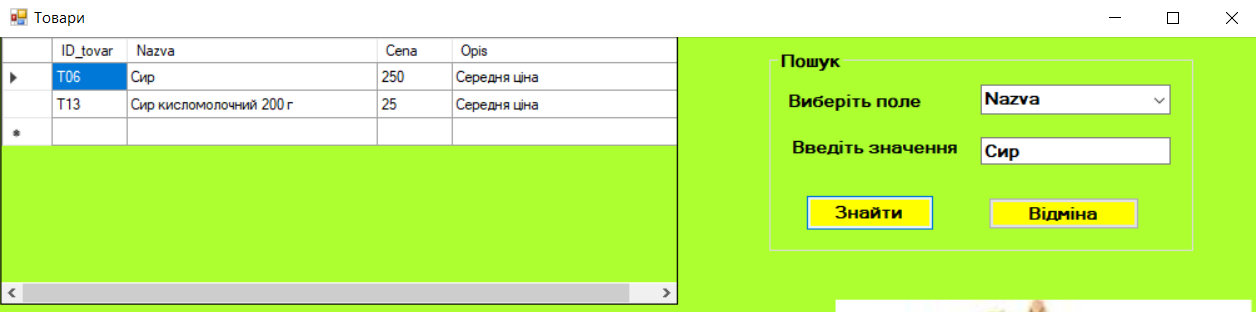


Рисунок 2.4 – Пошук конкретного товара

Нижче розміщені вікна для роботи з іншими таблицями (відповідно рисунки 2.5, 2.6, 2.7).

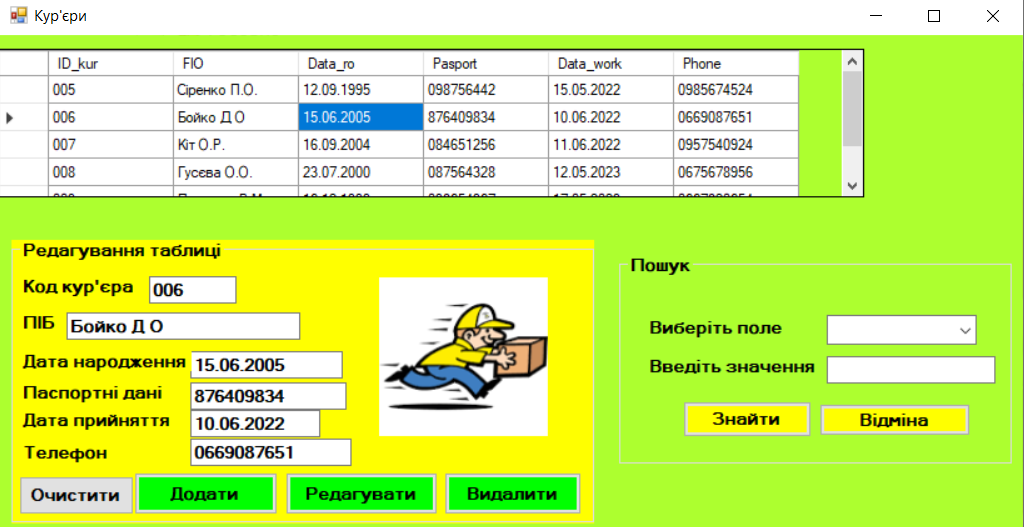


Рисунок 2.5 – Вигляд форми для роботи з таблицею Кур’єри

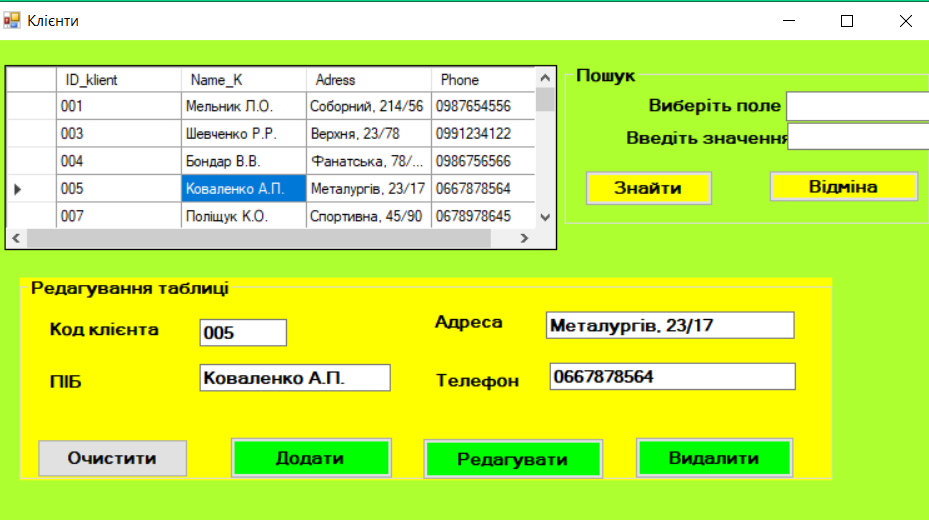


Рисунок 2.6 – Вигляд форми для роботи з таблицею «Клієнти»

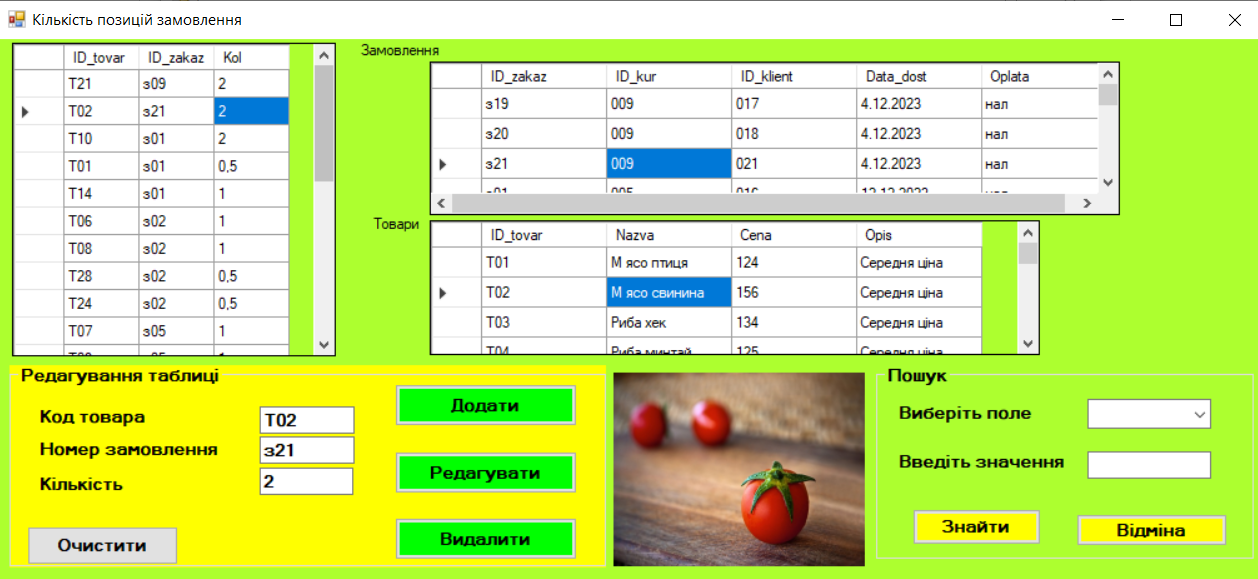


Рисунок 2.7 – Вигляд форми для роботи з таблицею

«Кількість позицій замовлення»

Замовлення оформлюється при натисненні на кнопку Замовлення (рисунок 2.8).

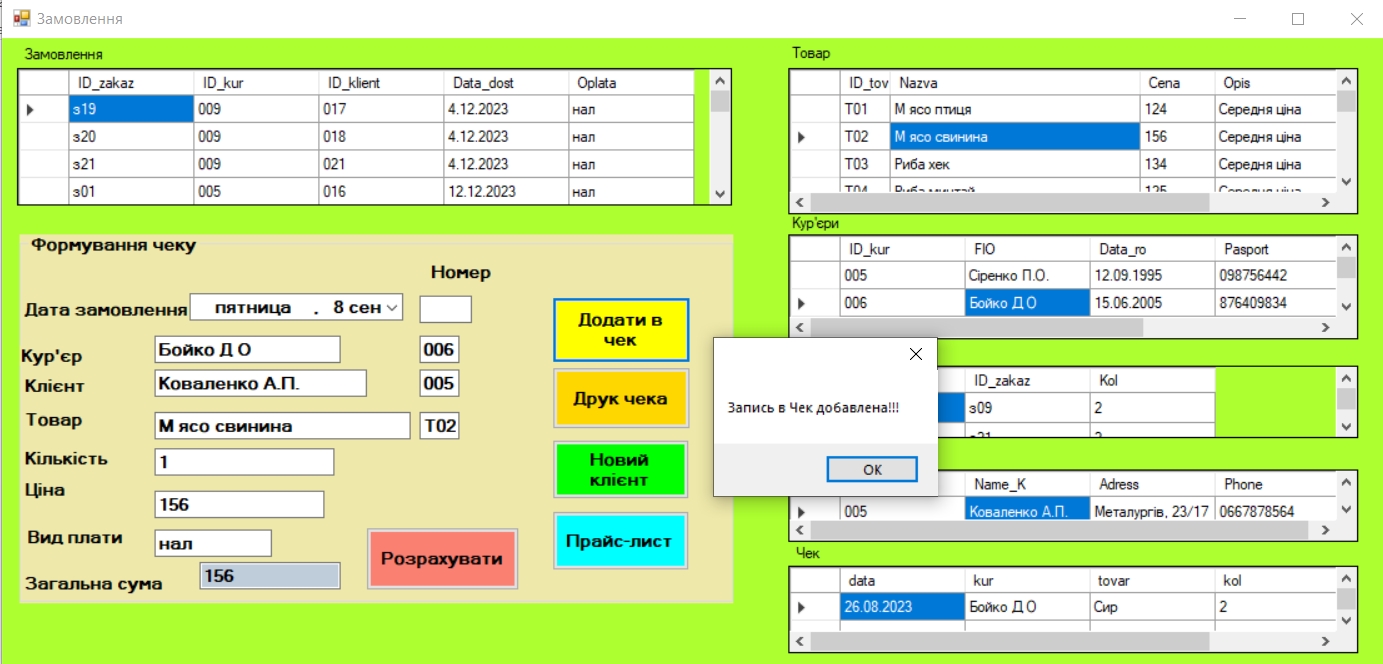


Рисунок 2.8 – Замовлення товару

Необхідно вибрати всі дані з допоміжних таблиць, натиснути кнопку «Розрахувати», а потім кнопку «Додати в чек», з’явиться відповідне повідомлення, і запис буде відображатися в таблиці Чек. Можна натиснути кнопку «Друк чека», сформується звіт (рисунок 2.9).

При натисненні на кнопку «Прайс-лист» сформується відповідний звіт, зображений на рисунку 2.10.

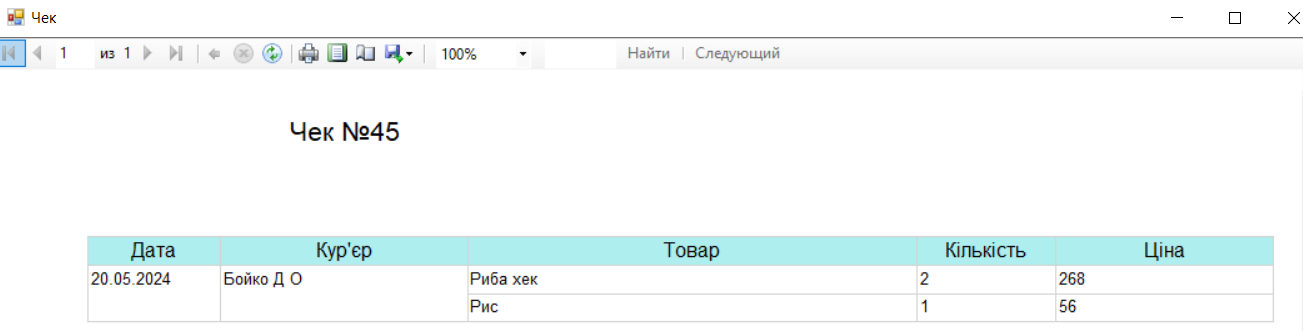


Рисунок 2.9 – Форма для друку чека

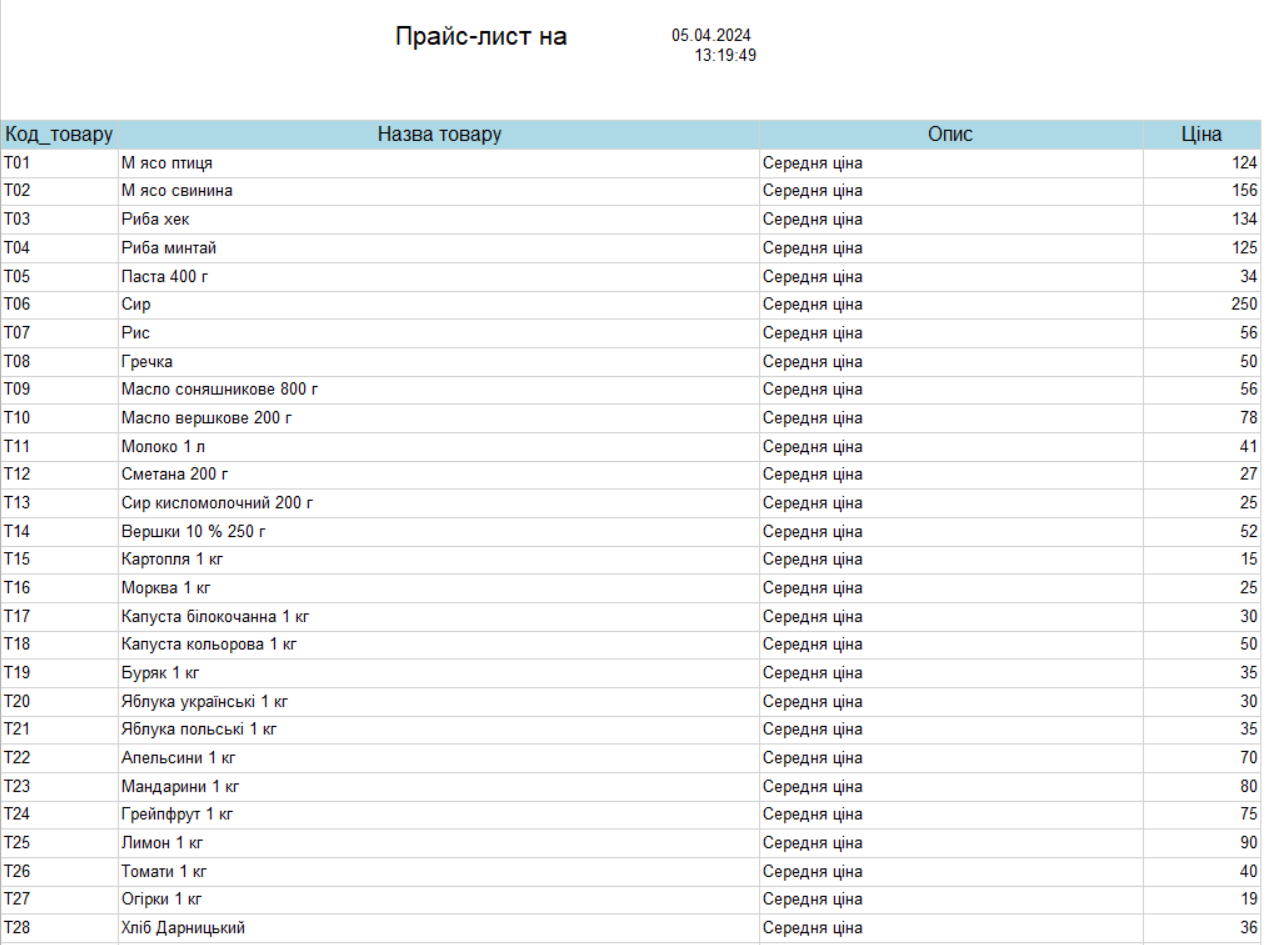


Рисунок 2.10 – Звіт «Прайс-лист»

**ВИСНОВКИ**

В результаті виконання курсової роботи була розроблена програма, призначена для автоматизації роботи з документами кур’єрської служби.

Програма виконує наступні функції:

* перегляд інформації про товари, клієнтів, кур’єрів;
* можливість додавання, редагування та видалення інформацію;
* виконання замовлення;
* формування чеку;
* пошук конкретної інформації;
* створення звітів.

Програмне забезпечення розроблене за допомогою візуального середовища розроботки Visual Studio 2010 та платформи .NET Framework 4.5.

Програмний продукт працює на персональному комп’ютері під керуванням операційних систем Windows 10/7. Програма має зручний, максимально орієнтований на кінцевого користувача інтерфейс.

Програма була протестована та виявилася працездатною, має достатньо велику швидкість роботи.

**ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ**

1

<https://stud.com.ua › ponyattya_avtomatizovani_sistemi_upravlinnya>

2 <https://www.work.ua/career-guide/courier/>

3 [https://pravnik.com.ua/forum/viewtopic.php](https://pravnik.com.ua/forum/viewtopic.php?f=16&t=344)

4 https://uk.wikipedia.org/wiki/Кур’єр

5 [Мова програмування C# 7.0 та платформи .NET та .NET Core, Філіп Джепікс, Ендрю Троєлсен](https://books.google.com.ua/books?id=Jus_DwAAQBAJ&printsec=frontcover&redir_esc=y#v=onepage&q&f=true) – «Діалектика», 2021

6 [Книга C# 4.0: повний посібник, Герберт Шилдт](https://www.amazon.com/4-0-Complete-Reference-Herbert-Schildt-ebook/dp/B003QP3XIM/ref=sr_1_1?crid=15D67V9VE0XUE&keywords=C%23+4.0+The+Complete+Reference+Herbert+Schildt&qid=1663588552&sprefix=c+4.0+the+complete+reference+herbert+schildt%2Caps%2C131&sr=8-1) - «Діалектика», 2019

7 Гайдаржи В. І., Ізварін І. В. Бази даних в інформаційних системах - [Університет "Україна"](https://profbook.com.ua/index.php?route=product/manufacturer/info&manufacturer_id=329), 2018

8 Г.А. Гурвиц. Microsoft Access 2010. – «Магнолія», 2010.

9 Байдачний С., Остапчук М. Windows 10 для C# розробників. - Київ : IT-книга, 2016

**ДОДАТОК А**

**(обов’язковий)**

**Текст програми**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using System.Data.OleDb;

namespace magazin

{

public partial class Form1 : Form

{

public int selectrow = -1;

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

string ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " + "Data Source=courier.mdb";

OleDbConnection connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

string sql = "SELECT \* FROM Tovar";

OleDbCommand myCommand = new OleDbCommand(sql, connection);

connection.Open();

OleDbDataAdapter da = new OleDbDataAdapter(myCommand);

DataSet ds = new DataSet();

da.Fill(ds, "Tovar");

dataGridView1.DataSource = ds.Tables["Tovar"].DefaultView;

dataGridView1.Columns[0].Width = 60;

dataGridView1.Columns[1].Width = 200;

dataGridView1.Columns[2].Width = 60;

dataGridView1.Columns[3].Width = 200;

connection.Close();

}

private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

selectrow = dataGridView1.CurrentCell.RowIndex;

if (selectrow < (dataGridView1.RowCount - 1))

{

//string num = Convert.ToString(selectrow);

//num = num + ".jpg";

//pictureBox1.Load(num);

textBox1.Text = "";

textBox2.Text = "";

textBox4.Text = "";

textBox5.Text = "";

textBox1.Text = dataGridView1[0, selectrow].Value.ToString();

textBox2.Text = dataGridView1[1, selectrow].Value.ToString();

textBox4.Text = dataGridView1[2, selectrow].Value.ToString();

textBox5.Text = dataGridView1[3, selectrow].Value.ToString();

}

}

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

{

// Add Tovar

string ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " + "Data Source=courier.mdb";

OleDbConnection connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

string sql;

string kod = textBox1.Text.ToString();

string nazva = textBox2.Text.ToString();

string cena = textBox4.Text.ToString();

string opis = textBox5.Text.ToString();

textBox1.Visible = false;

OleDbCommand myCommand;

sql = "INSERT INTO " + " Tovar (ID\_tovar, Nazva, Cena, Opis) " +" VALUES ("+"'" + kod + "'," + "'"+ nazva + "'," + cena + "," + "'"+opis+"'"+ ")";

//MessageBox.Show(sql);

myCommand = new OleDbCommand(sql, connection);

connection.Open();

myCommand.ExecuteNonQuery();

connection.Close();

//для перегляду результатів – створення нового з’єднання

connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

sql = "SELECT \* FROM Tovar";

myCommand = new OleDbCommand(sql, connection);

connection.Open();

OleDbDataAdapter da = new OleDbDataAdapter(myCommand);

DataSet ds = new DataSet();

da.Fill(ds, "Tovar");

dataGridView1.DataSource = ds.Tables["Tovar"].DefaultView;

connection.Close();

}

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

{

// Upd Tovar

string ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " + "Data Source=courier.mdb";

OleDbConnection connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

string sql = " UPDATE Tovar SET " +

" ID\_tovar = '" + textBox1.Text + "', " +

" Nazva = '" + textBox2.Text + "', " +

" Cena = " + textBox4.Text + ", " +

" Opis = '" + textBox5.Text +"'"+

" WHERE ID\_tovar = '" + textBox1.Text+"'";

MessageBox.Show(sql);

OleDbCommand myCommand = new OleDbCommand(sql, connection);

connection.Open();

myCommand.ExecuteNonQuery();

connection.Close();

connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

sql = "SELECT \* FROM Tovar";

myCommand = new OleDbCommand(sql, connection);

connection.Open();

OleDbDataAdapter da = new OleDbDataAdapter(myCommand);

DataSet ds = new DataSet();

da.Fill(ds, "Tovar");

dataGridView1.DataSource = ds.Tables["Tovar"].DefaultView;

connection.Close();

}

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

{

textBox1.Text = "";

textBox2.Text = "";

textBox4.Text = "";

textBox5.Text = "";

}

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

{

string ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " + "Data Source=courier.mdb";

OleDbConnection connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

string kod = textBox1.Text.ToString();

string sql = "DELETE \* FROM Tovar WHERE ID\_tovar = '" + kod+"'";

OleDbCommand myCommand = new OleDbCommand(sql, connection);

connection.Open();

myCommand.ExecuteNonQuery();

connection.Close();

connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

sql = "SELECT \* FROM Tovar";

myCommand = new OleDbCommand(sql, connection);

connection.Open();

OleDbDataAdapter da = new OleDbDataAdapter(myCommand);

DataSet ds = new DataSet();

da.Fill(ds, "Tovar");

dataGridView1.DataSource = ds.Tables["Tovar"].DefaultView;

connection.Close();

MessageBox.Show("Товар видалений!");

}

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

{

//Поиск товара

if (comboBox1.Text != "" && textBox7.Text != "")

{

string ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " + "Data Source=courier.mdb";

OleDbConnection connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

string znachenie = textBox7.Text.ToString();

string criteriy = comboBox1.SelectedItem.ToString();

string sql = "SELECT \* FROM [Tovar] WHERE " +

criteriy + " LIKE" + "'%" + znachenie + "%'";

OleDbCommand myCommand = new OleDbCommand(sql, connection);

connection.Open();

OleDbDataAdapter da = new OleDbDataAdapter(myCommand);

DataSet ds = new DataSet();

da.Fill(ds, "Tovar");

dataGridView1.DataSource = ds.Tables["Tovar"].DefaultView;

connection.Close();

}

else MessageBox.Show("Ви ввели не всі значения, заповніть поля для введення!");

}

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

{

//Отмена поиска

string ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " + "Data Source=courier.mdb";

OleDbConnection connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

string sql = "SELECT \* FROM Tovar";

OleDbCommand myCommand = new OleDbCommand(sql, connection);

connection.Open();

OleDbDataAdapter da = new OleDbDataAdapter(myCommand);

DataSet ds = new DataSet();

da.Fill(ds, "Tovar");

dataGridView1.DataSource = ds.Tables["Tovar"].DefaultView;

connection.Close();

}

private void textBox1\_TextChanged(object sender, EventArgs e)

{

toolTip1.Show("Для додавання запису введіть код і заповніть всі поля!", textBox1, textBox1.Location, 5000);

}

}

}

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using System.Data.OleDb;

namespace magazin

{

public partial class Form2 : Form

{

public int selectrow = -1;

public Form2()

{

InitializeComponent();

}

private void Form2\_Load(object sender, EventArgs e)

{

string ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " + "Data Source=courier.mdb";

OleDbConnection connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

string sql = "SELECT \* FROM kur";

OleDbCommand myCommand = new OleDbCommand(sql, connection);

connection.Open();

OleDbDataAdapter da = new OleDbDataAdapter(myCommand);

DataSet ds = new DataSet();

da.Fill(ds, "kur");

dataGridView1.DataSource = ds.Tables["kur"].DefaultView;

connection.Close();

}

private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

textBox1.Text = "";

textBox3.Text = "";

textBox4.Text = "";

textBox2.Text = "";

textBox5.Text = "";

textBox6.Text = "";

selectrow = dataGridView1.CurrentCell.RowIndex;

if (selectrow < (dataGridView1.RowCount - 1))

{

textBox1.Text = dataGridView1[0, selectrow].Value.ToString();

textBox2.Text = dataGridView1[1, selectrow].Value.ToString();

textBox3.Text = dataGridView1[2, selectrow].Value.ToString();

textBox4.Text = dataGridView1[3, selectrow].Value.ToString();

textBox5.Text = dataGridView1[4, selectrow].Value.ToString();

textBox6.Text = dataGridView1[5, selectrow].Value.ToString();

}

}

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

{

//Очистить

textBox1.Text = "";

textBox3.Text = "";

textBox4.Text = "";

textBox2.Text = "";

textBox5.Text = "";

textBox6.Text = "";

}

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

{

string ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " + "Data Source=courier.mdb";

OleDbConnection connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

string sql;

string kod = textBox1.Text.ToString();

string fio = textBox2.Text.ToString();

string dr = textBox3.Text.ToString();

string pas=textBox4.Text.ToString();

string dw = textBox5.Text.ToString();

string tel = textBox6.Text.ToString();

OleDbCommand myCommand;

sql = "INSERT INTO " + " kur (Id\_kur, FIO, data\_ro, pasport, data\_work, phone) " +

" VALUES (" + "'" + kod + "', "+ "'" + fio + "', " + "'" + dr + "'," +"'" + pas + "',"+ "'" + dw + "',"+ "'" + tel + "'"+")";

MessageBox.Show(sql);

myCommand = new OleDbCommand(sql, connection);

connection.Open();

myCommand.ExecuteNonQuery();

connection.Close();

//для перегляду результатів – створення нового з’єднання

connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

sql = "SELECT \* FROM kur";

myCommand = new OleDbCommand(sql, connection);

connection.Open();

OleDbDataAdapter da = new OleDbDataAdapter(myCommand);

DataSet ds = new DataSet();

da.Fill(ds, "kur");

dataGridView1.DataSource = ds.Tables["kur"].DefaultView;

connection.Close();

}

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

{

//Ред kur

string ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " + "Data Source=courier.mdb";

OleDbConnection connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

string kod = textBox1.Text.ToString();

string fio = textBox2.Text.ToString();

string dr = textBox3.Text.ToString();

string pas = textBox4.Text.ToString();

string dw = textBox5.Text.ToString();

string tel = textBox6.Text.ToString();

string sql = " UPDATE kur SET " + " ID\_kur = '" + kod + "', " + " FIO = '" +fio + "', "+ "data\_ro= '"+ dr + "', "+

" pasport = '" + pas + "', " + " data\_work = '" + dw + "', " + " phone = '" + tel + "' " +

" WHERE ID\_kur = " + "'"+kod+"'";

MessageBox.Show(sql);

OleDbCommand myCommand = new OleDbCommand(sql, connection);

connection.Open();

myCommand.ExecuteNonQuery();

connection.Close();

connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

sql = "SELECT \* FROM kur";

myCommand = new OleDbCommand(sql, connection);

connection.Open();

OleDbDataAdapter da = new OleDbDataAdapter(myCommand);

DataSet ds = new DataSet();

da.Fill(ds, "kur");

dataGridView1.DataSource = ds.Tables["kur"].DefaultView;

connection.Close();

}

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

{

//Удал kur

string ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " + "Data Source=courier.mdb";

OleDbConnection connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

string kod = textBox1.Text.ToString();

string sql = "DELETE \* FROM kur WHERE ID\_kur = " + "'"+kod+"'";

MessageBox.Show(sql);

OleDbCommand myCommand = new OleDbCommand(sql, connection);

connection.Open();

myCommand.ExecuteNonQuery();

connection.Close();

connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

sql = "SELECT \* FROM kur";

myCommand = new OleDbCommand(sql, connection);

connection.Open();

OleDbDataAdapter da = new OleDbDataAdapter(myCommand);

DataSet ds = new DataSet();

da.Fill(ds, "kur");

dataGridView1.DataSource = ds.Tables["kur"].DefaultView;

connection.Close();

}

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

{

//Поиск kur

if (comboBox1.Text != "" && textBox7.Text != "")

{

string ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " + "Data Source=courier.mdb";

OleDbConnection connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

string znachenie = textBox7.Text.ToString();

string criteriy = comboBox1.SelectedItem.ToString();

string sql = "SELECT \* FROM [kur] WHERE " + criteriy + " LIKE " + "'%" + znachenie + "%'";

//MessageBox.Show(sql);

OleDbCommand myCommand = new OleDbCommand(sql, connection);

connection.Open();

OleDbDataAdapter da = new OleDbDataAdapter(myCommand);

DataSet ds = new DataSet();

da.Fill(ds, "kur");

dataGridView1.DataSource = ds.Tables["kur"].DefaultView;

connection.Close();

}

else MessageBox.Show("Введіть всі значення");

}

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

{

//Отмена поиска

string ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " + "Data Source=courier.mdb";

OleDbConnection connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

string sql = "SELECT \* FROM kur";

OleDbCommand myCommand = new OleDbCommand(sql, connection);

connection.Open();

OleDbDataAdapter da = new OleDbDataAdapter(myCommand);

DataSet ds = new DataSet();

da.Fill(ds, "kur");

dataGridView1.DataSource = ds.Tables["kur"].DefaultView;

connection.Close();

}

}

}

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using System.Data.OleDb;

namespace magazin

{

public partial class Form3 : Form

{

public int selectrow = -1;

public Form3()

{

InitializeComponent();

}

private void Form3\_Load(object sender, EventArgs e)

{

string ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " + "Data Source=courier.mdb";

OleDbConnection connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

string sql = "SELECT \* FROM Klient";

OleDbCommand myCommand = new OleDbCommand(sql, connection);

connection.Open();

OleDbDataAdapter da = new OleDbDataAdapter(myCommand);

DataSet ds = new DataSet();

da.Fill(ds, "Klient");

dataGridView1.DataSource = ds.Tables["Klient"].DefaultView;

connection.Close();

}

private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

textBox1.Text = "";

textBox3.Text = "";

textBox4.Text = "";

textBox2.Text = "";

selectrow = dataGridView1.CurrentCell.RowIndex;

if (selectrow < (dataGridView1.RowCount - 1))

{

textBox1.Text = dataGridView1[0, selectrow].Value.ToString();

textBox3.Text = dataGridView1[1, selectrow].Value.ToString();

textBox4.Text = dataGridView1[2, selectrow].Value.ToString();

textBox2.Text = dataGridView1[3, selectrow].Value.ToString();

}

}

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

{

//Очистить

textBox1.Text = "";

textBox3.Text = "";

textBox4.Text = "";

textBox2.Text = "";

}

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

{

//Поиск

if (comboBox1.Text != "" && textBox7.Text != "")

{

string ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " + "Data Source=courier.mdb";

OleDbConnection connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

string znachenie = textBox7.Text.ToString();

string criteriy = comboBox1.SelectedItem.ToString();

string sql = "SELECT \* FROM [Klient] WHERE " + criteriy + " LIKE " + "'%" + znachenie + "%'";

//MessageBox.Show(sql);

OleDbCommand myCommand = new OleDbCommand(sql, connection);

connection.Open();

OleDbDataAdapter da = new OleDbDataAdapter(myCommand);

DataSet ds = new DataSet();

da.Fill(ds, "Klient");

dataGridView1.DataSource = ds.Tables["Klient"].DefaultView;

connection.Close();

}

else MessageBox.Show("Заповніть поля для введення");

}

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

{

//Отмена поиска

string ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " + "Data Source=courier.mdb";

OleDbConnection connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

string sql = "SELECT \* FROM Klient";

OleDbCommand myCommand = new OleDbCommand(sql, connection);

connection.Open();

OleDbDataAdapter da = new OleDbDataAdapter(myCommand);

DataSet ds = new DataSet();

da.Fill(ds, "Klient");

dataGridView1.DataSource = ds.Tables["Klient"].DefaultView;

connection.Close();

}

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

{

//доб

string ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " + "Data Source=courier.mdb";

OleDbConnection connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

string sql;

string kod=textBox1.Text.ToString();

string fio = textBox3.Text.ToString();

string adress = textBox4.Text.ToString();

string tel = textBox2.Text.ToString();

OleDbCommand myCommand;

sql = "INSERT INTO " + " klient (ID\_Klient, Name\_k, Adress, Phone) " +

" VALUES (" + "'" + kod + "',"+"'" + fio + "'," + "'" + adress + "'," + "'" + tel + "'"+")";

myCommand = new OleDbCommand(sql, connection);

connection.Open();

myCommand.ExecuteNonQuery();

connection.Close();

//для перегляду результатів – створення нового з’єднання

connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

sql = "SELECT \* FROM Klient";

myCommand = new OleDbCommand(sql, connection);

connection.Open();

OleDbDataAdapter da = new OleDbDataAdapter(myCommand);

DataSet ds = new DataSet();

da.Fill(ds, "Klient");

dataGridView1.DataSource = ds.Tables["Klient"].DefaultView;

connection.Close();

}

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

{

//ред

string ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " + "Data Source=courier.mdb";

OleDbConnection connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

string kod = textBox1.Text.ToString();

string fio = textBox3.Text.ToString();

string adress = textBox4.Text.ToString();

string tel = textBox2.Text.ToString();

string sql = " UPDATE Klient SET " + " ID\_Klient = '" + kod + "', "+" Name\_K = '" + fio + "', " + " Adress = '" + adress + "', "

+ " Phone = '" + tel + "' "+" WHERE ID\_klient = " + "'"+kod+"'";

MessageBox.Show(sql);

OleDbCommand myCommand = new OleDbCommand(sql, connection);

connection.Open();

myCommand.ExecuteNonQuery();

connection.Close();

connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

sql = "SELECT \* FROM Klient";

myCommand = new OleDbCommand(sql, connection);

connection.Open();

OleDbDataAdapter da = new OleDbDataAdapter(myCommand);

DataSet ds = new DataSet();

da.Fill(ds, "Klient");

dataGridView1.DataSource = ds.Tables["Klient"].DefaultView;

connection.Close();

}

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

{

//удал

string ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " + "Data Source=courier.mdb";

OleDbConnection connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

string kod = textBox1.Text.ToString();

string sql = "DELETE \* FROM courier WHERE ID\_klient = " + kod;

OleDbCommand myCommand = new OleDbCommand(sql, connection);

connection.Open();

myCommand.ExecuteNonQuery();

connection.Close();

connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

sql = "SELECT \* FROM Klient";

myCommand = new OleDbCommand(sql, connection);

connection.Open();

OleDbDataAdapter da = new OleDbDataAdapter(myCommand);

DataSet ds = new DataSet();

da.Fill(ds, "Klient");

dataGridView1.DataSource = ds.Tables["Klient"].DefaultView;

connection.Close();

MessageBox.Show("Клієнт видалений!");

}

private void textBox1\_TextChanged(object sender, EventArgs e)

{

toolTip1.Show("Заповніть всі поля!", textBox1, textBox1.Location, 5000);

}

}

}

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using System.Data.OleDb;

namespace magazin

{

public partial class Form4 : Form

{

public int selectrow = -1;

public Form4()

{

InitializeComponent();

}

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

{

string ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " + "Data Source=courier.mdb";

OleDbConnection connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

string sql = "SELECT \* FROM Zakaz";

OleDbCommand myCommand = new OleDbCommand(sql, connection);

connection.Open();

OleDbDataAdapter da = new OleDbDataAdapter(myCommand);

DataSet ds = new DataSet();

da.Fill(ds, "Zakaz");

dataGridView1.DataSource = ds.Tables["Zakaz"].DefaultView;

connection.Close();

string sq2 = "SELECT \* FROM Tovar";

OleDbCommand myCommand1 = new OleDbCommand(sq2, connection);

connection.Open();

OleDbDataAdapter da1 = new OleDbDataAdapter(myCommand1);

DataSet ds1 = new DataSet();

da1.Fill(ds1, "Tovar");

dataGridView2.DataSource = ds1.Tables["Tovar"].DefaultView;

connection.Close();

dataGridView2.Columns[0].Width = 40;

dataGridView2.Columns[1].Width = 200;

dataGridView2.Columns[2].Width = 60;

dataGridView2.Columns[3].Width = 200;

string sq3 = "SELECT \* FROM kur";

OleDbCommand myCommand2 = new OleDbCommand(sq3, connection);

connection.Open();

OleDbDataAdapter da2 = new OleDbDataAdapter(myCommand2);

DataSet ds2 = new DataSet();

da2.Fill(ds2, "kur");

dataGridView3.DataSource = ds2.Tables["kur"].DefaultView;

connection.Close();

string sq4 = "SELECT \* FROM klient";

OleDbCommand myCommand4 = new OleDbCommand(sq4, connection);

connection.Open();

OleDbDataAdapter da4 = new OleDbDataAdapter(myCommand4);

DataSet ds4 = new DataSet();

da4.Fill(ds4, "klient");

dataGridView6.DataSource = ds4.Tables["klient"].DefaultView;

connection.Close();

string sq5 = "SELECT \* FROM kol\_position";

OleDbCommand myCommand3= new OleDbCommand(sq5, connection);

connection.Open();

OleDbDataAdapter da3 = new OleDbDataAdapter(myCommand3);

DataSet ds3 = new DataSet();

da3.Fill(ds3, "kol\_position");

dataGridView5.DataSource = ds3.Tables["kol\_position"].DefaultView;

connection.Close();

string ConnectionString1 = "Provider=Microsoft.Jet.OLEDB.4.0; " + "Data Source=courier.mdb";

OleDbConnection connection1 = new OleDbConnection();

connection1.ConnectionString = ConnectionString1;

string sq7 = "SELECT \* FROM Чек";

OleDbCommand myCommand7 = new OleDbCommand(sq7, connection1);

connection.Open();

OleDbDataAdapter da7 = new OleDbDataAdapter(myCommand7);

DataSet ds7 = new DataSet();

da7.Fill(ds7, "Чек");

dataGridView4.DataSource = ds7.Tables["Чек"].DefaultView;

connection.Close();

}

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

{

dataGridView4.Visible = true;

//Add Check

string ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " + "Data Source=courier.mdb";

OleDbConnection connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

string sql;

string data = dateTimePicker1.Value.ToString("dd.MM.yyyy");

// MessageBox.Show(data);

string work = textBox1.Text.ToString();

string tovar = textBox2.Text.ToString();

string kol = textBox3.Text.ToString();

string stoim = textBox5.Text.ToString();

string kod = textBox6.Text.ToString();

string kods = textBox7.Text.ToString();

string id\_zakaz= textBox10.Text.ToString();

string kodk = textBox9.Text.ToString();

string namek = textBox8.Text.ToString();

string opl=textBox11.Text.ToString();

string cena = textBox4.Text.ToString();

//(Номер, Дата, Вид, Сотрудник, Товар, Количество)

sql = "INSERT INTO Чек (data, kur, tovar, kol, stoim) " +

" values ( " + "'" + data + "'," + "'" + work + "'," + "'" + tovar + "'," + "'"+kol + "'," + "'"+stoim +"'"+ ")";

MessageBox.Show(sql);

OleDbCommand myCommand = new OleDbCommand(sql, connection);

connection.Open();

myCommand.ExecuteNonQuery();

connection.Close();

MessageBox.Show("Запись в Чек добавлена!!!");

//для перегляду результатів – створення нового з’єднання

connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

string sql1 = "SELECT \* FROM Чек";

myCommand = new OleDbCommand(sql1, connection);

connection.Open();

OleDbDataAdapter da = new OleDbDataAdapter(myCommand);

DataSet ds = new DataSet();

da.Fill(ds, "Чек");

dataGridView4.DataSource = ds.Tables["Чек"].DefaultView;

connection.Close();

}

private void dataGridView2\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{//Tovar

textBox2.Text = "";

textBox4.Text = "";

textBox6.Text = "";

selectrow = dataGridView2.CurrentCell.RowIndex;

if (selectrow < (dataGridView2.RowCount - 1))

{

textBox2.Text = dataGridView2[1, selectrow].Value.ToString();

textBox4.Text = dataGridView2[2, selectrow].Value.ToString();

textBox6.Text = dataGridView2[0, selectrow].Value.ToString();

}

}

private void dataGridView3\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{//KUR

textBox1.Text = "";

textBox7.Text = "";

selectrow = dataGridView3.CurrentCell.RowIndex;

if (selectrow < (dataGridView3.RowCount - 1))

{

textBox1.Text = dataGridView3[1, selectrow].Value.ToString();

textBox7.Text = dataGridView3[0, selectrow].Value.ToString();

}

}

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

{

double st = Convert.ToDouble(textBox3.Text) \* Convert.ToDouble(textBox4.Text);

textBox5.Text = Convert.ToString(st);

}

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

{

string ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " + "Data Source=courier.mdb";

OleDbConnection connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

string sql = "DELETE \* FROM Чек";

//MessageBox.Show(sql);

OleDbCommand myCommand = new OleDbCommand(sql, connection);

connection.Open();

myCommand.ExecuteNonQuery();

connection.Close();

textBox1.Text = "";

textBox2.Text = "";

textBox3.Text = "";

textBox5.Text = "";

textBox4.Text = "";

textBox6.Text = "";

dataGridView4.Visible = false;

}

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

{

Form6 f6 = new Form6();

f6.Show();

}

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

{

Form7 f7 = new Form7();

f7.Show();

}

private void dataGridView6\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

//Klient

textBox8.Text = "";

textBox9.Text = "";

selectrow = dataGridView6.CurrentCell.RowIndex;

if (selectrow < (dataGridView6.RowCount - 1))

{

textBox8.Text = dataGridView6[1, selectrow].Value.ToString();

textBox9.Text = dataGridView6[0, selectrow].Value.ToString();

}

}

private void dataGridView5\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

//Kol

textBox10.Text = "";

textBox3.Text = "";

selectrow = dataGridView5.CurrentCell.RowIndex;

if (selectrow < (dataGridView5.RowCount - 1))

{

textBox10.Text = dataGridView5[1, selectrow].Value.ToString();

textBox3.Text = dataGridView5[2, selectrow].Value.ToString();

}

}

}

}

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using System.Data.OleDb;

namespace magazin

{

public partial class Form5 : Form

{

public int selectrow = -1;

public Form5()

{

InitializeComponent();

}

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

{

string ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " + "Data Source=courier.mdb";

OleDbConnection connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

string sql = "SELECT \* FROM Kol\_position";

OleDbCommand myCommand = new OleDbCommand(sql, connection);

connection.Open();

OleDbDataAdapter da = new OleDbDataAdapter(myCommand);

DataSet ds = new DataSet();

da.Fill(ds, "Kol\_position");

dataGridView1.DataSource = ds.Tables["Kol\_position"].DefaultView;

connection.Close();

dataGridView1.Columns[0].Width = 60;

dataGridView1.Columns[1].Width = 60;

dataGridView1.Columns[2].Width = 60;

string sql1 = "SELECT \* FROM zakaz";

OleDbCommand myCommand1 = new OleDbCommand(sql1, connection);

connection.Open();

OleDbDataAdapter da1 = new OleDbDataAdapter(myCommand1);

DataSet ds1 = new DataSet();

da1.Fill(ds1, "zakaz");

dataGridView2.DataSource = ds1.Tables["zakaz"].DefaultView;

connection.Close();

string sql2 = "SELECT \* FROM tovar";

OleDbCommand myCommand2 = new OleDbCommand(sql2, connection);

connection.Open();

OleDbDataAdapter da2 = new OleDbDataAdapter(myCommand2);

DataSet ds2 = new DataSet();

da2.Fill(ds2, "tovar");

dataGridView3.DataSource = ds2.Tables["tovar"].DefaultView;

connection.Close();

}

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

{

textBox1.Text = "";

textBox3.Text = "";

textBox2.Text = "";

}

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

{

//Add

string ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " + "Data Source=courier.mdb";

OleDbConnection connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

string sql;

string kodt = textBox1.Text.ToString();

string kodz = textBox3.Text.ToString();

string kol = textBox2.Text.ToString();

OleDbCommand myCommand;

sql = "INSERT INTO " + "Kol\_position (ID\_tovar, ID\_zakaz, kol) " + " VALUES (" + "'" + kodt + "'," + "'" + kodz + "'," +kol+ ")";

myCommand = new OleDbCommand(sql, connection);

connection.Open();

myCommand.ExecuteNonQuery();

connection.Close();

//для перегляду результатів – створення нового з’єднання

connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

sql = "SELECT \* FROM Kol\_position";

myCommand = new OleDbCommand(sql, connection);

connection.Open();

OleDbDataAdapter da = new OleDbDataAdapter(myCommand);

DataSet ds = new DataSet();

da.Fill(ds, "Kol\_position");

dataGridView1.DataSource = ds.Tables["Kol\_position"].DefaultView;

connection.Close();

}

private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

textBox1.Text = "";

textBox3.Text = "";

textBox2.Text = "";

selectrow = dataGridView1.CurrentCell.RowIndex;

if (selectrow < (dataGridView1.RowCount - 1))

{

textBox1.Text = dataGridView1[0, selectrow].Value.ToString();

textBox3.Text = dataGridView1[1, selectrow].Value.ToString();

textBox2.Text = dataGridView1[2, selectrow].Value.ToString();

}

}

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

{

//Upd

string ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " + "Data Source=courier.mdb";

OleDbConnection connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

string kodt = textBox1.Text.ToString();

string kodz = textBox3.Text.ToString();

string kol = textBox2.Text.ToString();

string sql = " UPDATE Kol\_position SET " + " ID\_tovar = '" + kodt + "'," + " ID\_zakaz= '"+kodz+ "',"+ " kol="+ kol+

" WHERE ID\_tovar = " + kodt;

MessageBox.Show(sql);

OleDbCommand myCommand = new OleDbCommand(sql, connection);

connection.Open();

myCommand.ExecuteNonQuery();

connection.Close();

connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

sql = "SELECT \* FROM Kol\_position";

myCommand = new OleDbCommand(sql, connection);

connection.Open();

OleDbDataAdapter da = new OleDbDataAdapter(myCommand);

DataSet ds = new DataSet();

da.Fill(ds, "Kol\_position");

dataGridView1.DataSource = ds.Tables["Kol\_position"].DefaultView;

connection.Close();

}

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

{

//Del Tip

string ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " + "Data Source=courier.mdb";

OleDbConnection connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

string kodt = textBox1.Text.ToString();

string sql = "DELETE \* FROM Kol\_position WHERE ID\_tovar = " + kodt;

OleDbCommand myCommand = new OleDbCommand(sql, connection);

connection.Open();

myCommand.ExecuteNonQuery();

connection.Close();

connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

sql = "SELECT \* FROM Kol\_position";

myCommand = new OleDbCommand(sql, connection);

connection.Open();

OleDbDataAdapter da = new OleDbDataAdapter(myCommand);

DataSet ds = new DataSet();

da.Fill(ds, "Kol\_position");

dataGridView1.DataSource = ds.Tables["Kol\_position"].DefaultView;

connection.Close();

}

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

{

//Poisk

if (textBox7.Text != "")

{

string ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " + "Data Source=courier.mdb";

OleDbConnection connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

string znachenie = textBox7.Text.ToString();

string criteriy = comboBox1.SelectedItem.ToString();

string sql = "SELECT \* FROM [Kol\_position] WHERE "+ criteriy+ " LIKE " + "'%" + znachenie + "%'";

MessageBox.Show(sql);

OleDbCommand myCommand = new OleDbCommand(sql, connection);

connection.Open();

OleDbDataAdapter da = new OleDbDataAdapter(myCommand);

DataSet ds = new DataSet();

da.Fill(ds, "Kol\_position");

dataGridView1.DataSource = ds.Tables["Kol\_position"].DefaultView;

connection.Close();

}

else MessageBox.Show("Ви ввели невірне значення");

}

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

{

//Otmena poisk

string ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " + "Data Source=courier.mdb";

OleDbConnection connection = new OleDbConnection();

connection.ConnectionString = ConnectionString;

string sql = "SELECT \* FROM Kol\_position";

OleDbCommand myCommand = new OleDbCommand(sql, connection);

connection.Open();

OleDbDataAdapter da = new OleDbDataAdapter(myCommand);

DataSet ds = new DataSet();

da.Fill(ds, "Kol\_position");

dataGridView1.DataSource = ds.Tables["Kol\_position"].DefaultView;

connection.Close();

}

private void textBox1\_TextChanged(object sender, EventArgs e)

{

//toolTip1.Show("Для добавления записи код не вводится!", textBox1, textBox1.Location, 5000);

}

private void dataGridView2\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

//номер замовлення

textBox3.Text = "";

selectrow = dataGridView2.CurrentCell.RowIndex;

if (selectrow < (dataGridView2.RowCount - 1))

{

textBox3.Text = dataGridView2[0, selectrow].Value.ToString();

}

}

private void dataGridView3\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

//код товара

textBox1.Text = "";

selectrow = dataGridView3.CurrentCell.RowIndex;

if (selectrow < (dataGridView3.RowCount - 1))

{

textBox1.Text = dataGridView3[0, selectrow].Value.ToString();

}

}

}

}

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace magazin

{

public partial class Form6 : Form

{

public Form6()

{

InitializeComponent();

}

private void Form6\_Load(object sender, EventArgs e)

{

// TODO: данная строка кода позволяет загрузить данные в таблицу "courierDataSet3.Чек". При необходимости она может быть перемещена или удалена.

this.ЧекTableAdapter.Fill(this.courierDataSet3.Чек);

this.reportViewer1.RefreshReport();

}

}

}

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace magazin

{

public partial class Form7 : Form

{

public Form7()

{

InitializeComponent();

}

private void Form7\_Load(object sender, EventArgs e)

{

// TODO: данная строка кода позволяет загрузить данные в таблицу "courierDataSet.Tovar". При необходимости она может быть перемещена или удалена.

this.TovarTableAdapter.Fill(this.courierDataSet.Tovar);

// TODO: данная строка кода позволяет загрузить данные в таблицу "magazinDataSet2.Товар". При необходимости она может быть перемещена или удалена.

this.ТоварTableAdapter.Fill(this.magazinDataSet2.Товар);

this.reportViewer1.RefreshReport();

this.reportViewer1.RefreshReport();

}

}

}

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace magazin

{

public partial class FormMain : Form

{

public FormMain()

{

InitializeComponent();

}

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

{

Form1 f1 = new Form1();

f1.Show();

}

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

{

Close();

}

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

{

Form3 f3 = new Form3();

f3.Show();

}

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

{

Form2 f2 = new Form2();

f2.Show();

}

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

{

Form4 f4 = new Form4();

f4.Show();

}

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

{

Form5 f5 = new Form5();

f5.Show();

} }}

**ДОДАТОК Б**

**(обов’язковий)**

**Приклади звітів**

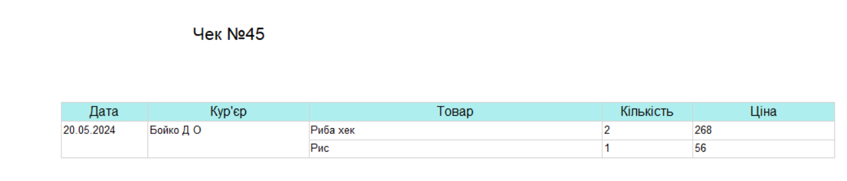
****

Рисунок Б.1 – Звіт «Чек»

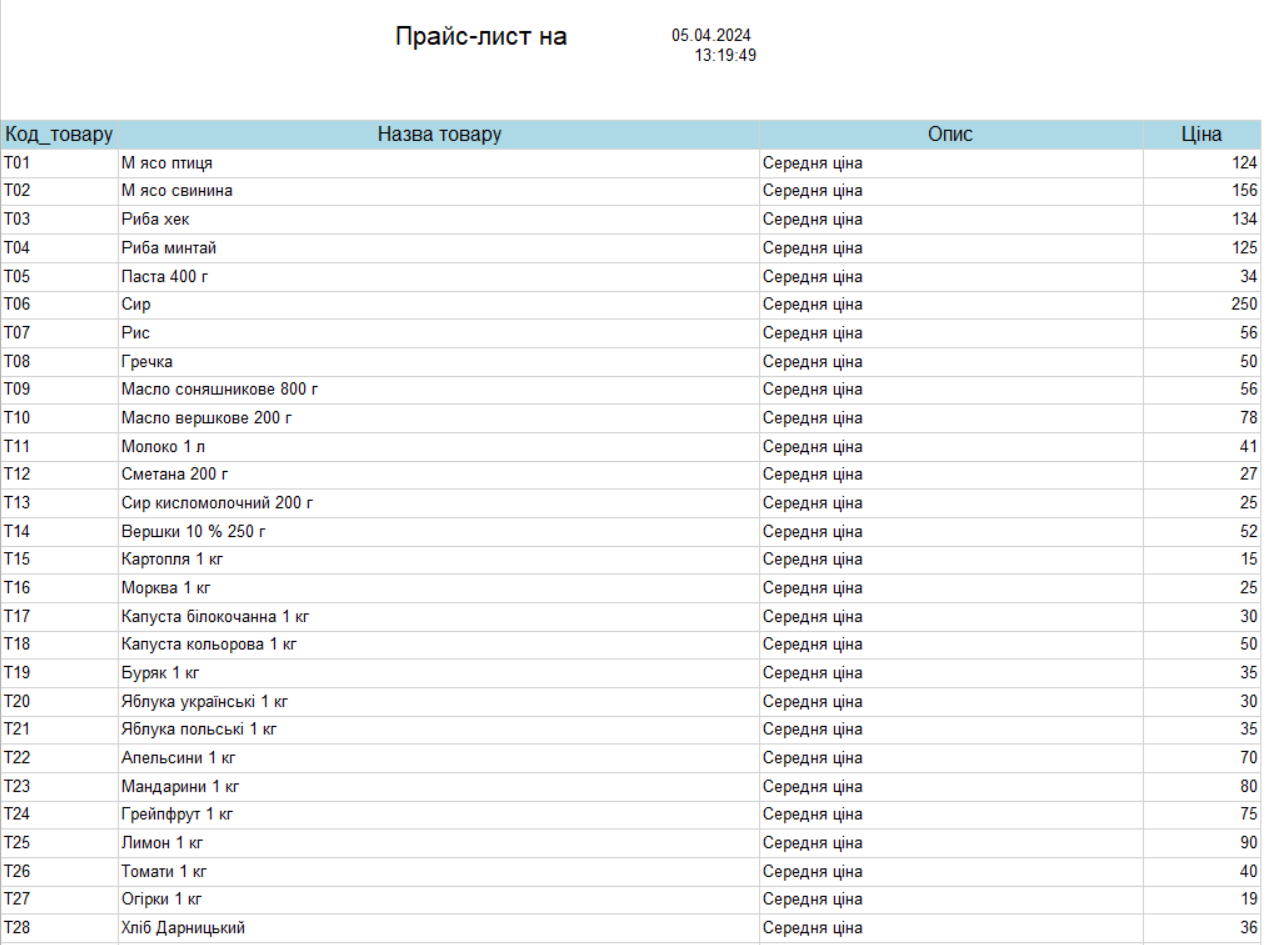
****

Рисунок Б.2 – Звіт «Прайс-лист»