

Державний торговельно-економічний університет

Кафедра комп'ютерних наук та інформаційних систем

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Інформаційна система управління діяльністю комунального підприємства»

Студента 4 курсу, 8 групи,

спеціальності

122 «Комп'ютерні науки»

Дорошенко Ігор

Артемович

*підпис
студента*

Науковий керівник
доктор фізико-математичних наук,
професор

Пурський Олег
Іванович

*підпис
керівника*

Гарант освітньої програми
кандидат технічних наук, доцент

Демідов Павло
Георгійович

*підпис
керівника*

Київ 2022

Державний торговельно-економічний університет

Факультет інформаційних технологій
Кафедра комп'ютерних наук та інформаційних систем
Спеціальність 122 «Комп'ютерні науки»

Затверджую

Зав. кафедри _____ Пурський О.І.
2022р.

Завдання **на випускню кваліфікаційну роботу (проект) студенту**

Дорошенко Ігор Артемович
(прізвище, ім'я, по батькові)

1. Тема випускної кваліфікаційної роботи (проекту)
Інформаційна система управління діяльністю комунального підприємства
Затверджена наказом ректора від
2. Строк здачі студентом закінченої роботи
3. Цільова установка та вихідні дані до роботи
Мета роботи: розробка інформаційної системи для адміністрування діяльності підприємства комунального зразку
Об'єкт дослідження: автоматизація процесів управління комунальним підприємством.
Предмет дослідження: інформаційна система управління та адміністрування процесами на підприємствах
4. Перелік графічного матеріалу _____

5. Консультанти по роботі із зазначенням розділів, за якими здійснюється консультування:

Розділ	Консультант (прізвище, ініціали)	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Пурський О. І.	7.12.2021 р.	7.12.2021 р.
2	Пурський О. І.	7.12.2021 р.	7.12.2021 р.
3	Пурський О. І..	7.12.2021 р.	7.12.2021 р.

6. Зміст випускної кваліфікаційної роботи (проекту) (перелік питань за кожним розділом)

Вступ

1. АНАЛІЗ ТЕОРЕТИКО-ПРАКТИЧНИХ ПІДХОДІВ ДО ВИРІШЕННЯ ЗАДАЧІ УПРАВЛІННЯ ПІДПРИЄМСТВОМ

1.1. Сучасний стан програмних рішень в області автоматизації управління підприємством

1.2. Характеристика програмних засобів та засобів створення та ведення баз даних

2. ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ УПРАВЛІННЯ ДІЯЛЬНІСТЮ КОМУНАЛЬНОГО ПІДПРИЄМТВА

2.1. Постановка задачі на розробку ІС

2.2. Проектування логічної моделі даних та нормалізація відношень ІС

2.3. Проектування фізичної моделі даних ІС

3. РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ УПРАВЛІННЯ ДІЯЛЬНІСТЮ КОМУНАЛЬНОГО ПІДПРИЄМТВА

3.1. Загальна архітектура ІС

3.2. Вхідні та вихідні форми ІС

3.3. Алгоритми роботи ІС та інформаційні потоки

ВИСНОВКИ

3.3. Інструкція користувача

3.4. Характеристика задіяних засобів розробки програмного забезпечення та прикладних програм ІС

ВИСНОВКИ

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

ДОДАТКИ

7. Календарний план виконання роботи

№ пор.	Назва етапів випускного кваліфікаційного проекту	Строк виконання етапів роботи	
		За планом	фактично
1	2	3	4
1	<i>Вибір теми випускного кваліфікаційного проекту</i>	01.11.2021	01.11.2021
2	<i>Розробка та затвердження завдання на випускний кваліфікаційний проект</i>	07.12.2021	07.12.2021
3	<i>Вступ</i>	03.02.2022	03.02.2022
4	<i>Розділ 1. Аналіз теоретико-практичних підходів до вирішення задачі управління підприємством</i>	25.02.2022	25.02.2022
5	<i>Розділ 2. Проектування інформаційної системи управління діяльністю комунального підприємства</i>	15.04.2022	15.04.2022
6	<i>Розділ 3. Розробка ІС управління діяльністю комунального підприємства</i>	26.05.2022	26.05.2022
7	<i>Висновки</i>	31.05.2022	31.05.2022
8	<i>Здача випускного кваліфікаційного проекту на кафедрі науковому керівнику</i>	10.06.2022	10.06.2022
9	<i>Попередній захист випускного кваліфікаційного проекту</i>	13.06.2022	13.06.2022
10	<i>Виправлення зауважень, зовнішнє рецензування випускного кваліфікаційного проекту</i>	15.06.2022	15.06.2022
12	<i>Представлення готового зшитого випускного кваліфікаційного проекту на кафедрі</i>	17.06.2022	17.06.2022
13	<i>Публічний захист випускного кваліфікаційного проекту</i>	За розкладом роботи ЕК	20.06.2022

8. Дата видачі завдання « » 2021 р.

9. Керівник випускної кваліфікаційної роботи (проекту)

Пурський О. І.

(прізвище, ініціали, підпис)

10. Гарант освітньої програми

Демідов П.Г.

(прізвище, ініціали, підпис)

11. Завдання прийняв до виконання студент-дипломник

Дорошенко І. А.

(прізвище, ініціали, підпис)



Анотація

У випускній кваліфікаційній роботі запропонована програма реалізує інформаційну систему управління діяльністю комунального підприємства, а саме реалізує довідкову систему, призначену для організації обліку розподілу робіт по відділах і працівникам, а також веде облік вартості виконаних робіт. Додаток дозволяє проводити введення, редагування і перегляд вмісту баз даних, а також відповідати на запити користувача і складати різноманітні звіти.

Отримані результати, можуть бути використані в різних організаціях, в тому числі і комунального підприємства. Дана програма легка в користуванні, дозволяє зберігати велику кількість відомостей в одній базі даних, економить робочий час за рахунок автоматизації деяких процесів таких як, облік вартості робіт, а також облік розподілу робіт по відділах. Тому запропонована програма повинна істотно спростити роботу.

Ключові слова: інформаційна система, комунальне підприємство, автоматизація, документообіг, бази даних, інформаційні потоки

Anotation

In the final qualifying work, the proposed program implements an information reference system designed to organize the accounting of the distribution of work by departments and employees, as well as records the cost of work performed. The application allows you to enter, edit and view the contents of databases, as well as respond to user requests and compile various reports.

The results obtained can be used in various organizations, including the format of LLC. This program is easy to use, allows you to store a large amount of information in one database, saves time by automating some processes such as accounting for the cost of work, as well as accounting for the distribution of work by department. Therefore, the proposed program should significantly simplify the work.

Keywords: information system, municipal enterprise, automation, document flow, databases, information flows.

ЗМІСТ

Contents

ВСТУП	9
РОЗДІЛ 1. АНАЛІЗ ТЕОРЕТИКО-ПРАКТИЧНИХ ПІДХОДІВ ДО ВИРІШЕННЯ ЗАДАЧІ УПРАВЛІННЯ ПІДПРИЄМСТВОМ	12
1.1 Сучасний стан програмних рішень в області автоматизації управління підприємством.....	12
1.2. Характеристика програмних засобів та засобів створення та ведення баз даних.....	15
РОЗДІЛ 2. ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ УПРАВЛІННЯ ДІЯЛЬНІСТЮ КОМУНАЛЬНОГО ПІДПРИЄМТВА	21
2.1 Постановка задачі на розробку ІС.....	21
2.2 Проектування логічної моделі даних.....	25
2.3 Проектування фізичної моделі даних ІС.....	28
РОЗДІЛ 3. РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ УПРАВЛІННЯ ДІЯЛЬНІСТЮ КОМУНАЛЬНОГО ПІДПРИЄМТВА	33
3.1. Загальна архітектура ІС.....	33
3.2. Вхідні та вихідні форми ІС.....	36
3.3 Алгоритми роботи ІС.....	36
3.4. Інструкція користувача.....	37
3.5 Характеристика задіяних засобів розробки програмного забезпечення та прикладних програм ІС.....	41
ВИСНОВОК	45
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	46
ДОДАТОК	49
Програмний код реалізації додатку.....	49

ВСТУП

У наш час електронні обчислювальні машини широко застосовуються в багатьох галузях діяльності людини. Ні одна установа не може обійтись у своїй роботі без використання комп'ютерів, які з успіхом замінюють рутинну роботу, яка виконувалась раніше в ручну, підвищуючи ефективність роботи будь-якого підприємства [23].

Сфера послуг в даний час є однією з найважливіших галузей народного господарства, створеної для задоволення індивідуальних запитів і потреб населення країни в різних її видах. Сфера послуг як галузь економічної діяльності являє собою сукупність організацій, мета яких - надання різноманітних платних послуг за індивідуальними замовленнями населення. Таким чином, сфера послуг вирішує найважливіші соціально-економічні завдання і її значення в житті суспільства стрімко зростає. Одним із видів таких послуг є послуги з надання авторемонтних послуг [14].

Комунальні Підприємства, знаходяться в умовах високої конкуренції. Щоб підвищити свою конкурентоспроможність більшість підприємств використовують інформаційні системи в процесі своєї роботи, так як обчислювальна техніка здатна в разі прискорити процес обробки інформації і отримання результату. Але в наші дні залишаються і такі підприємства, керівники яких не вирішуються впроваджувати інформаційні системи в силу їх високої ціни або своєї необізнаності в даній сфері, що і зумовило **актуальність** обраної теми дослідження, її мету та завдання.

Мета і завдання дослідження. Метою даного дослідження є розробка інформаційної системи для адміністрування діяльності підприємства комунального зразку. Для досягнення поставленої мети необхідно було вирішити наступні **завдання:**

- провести аналітичне дослідження проблематики організацій, що займається спільною по характеру діяльністю
- дослідити сучасний стан програмних рішень у сфері управління комунальними підприємствами
- дослідити методи автоматизації управління комунальними підприємствами
- розробити характеристику програмних засобів і засобів створення та ведення баз даних
- розробити логічну та фізичну модель даних інформаційної системи комунального підприємства
- розробити метод забезпечення визначення цін та знижок на виконані роботи комунального підприємства
- розробити інформаційну систему ведення документів з виконання робіт окомунального підприємства
- охарактеризувати задіяні засоби розробки програмного забезпечення

Об'єкт дослідження: автоматизація процесів управління комунальним підприємством.

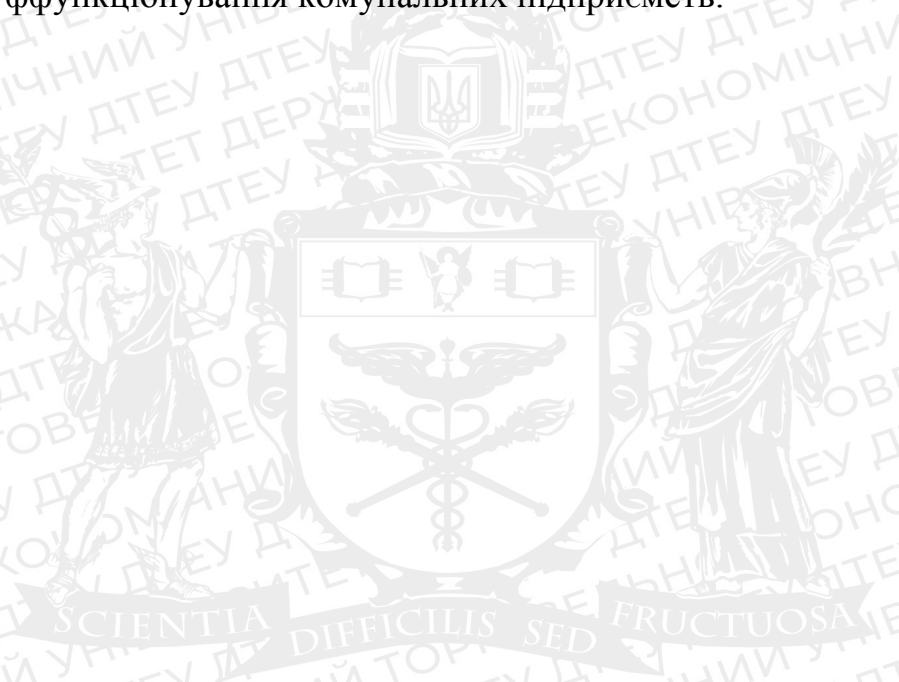
Предмет дослідження: інформаційна система управління та адміністрування процесами на підприємствах.

Методи дослідження: Теоретичною основою дослідження є загальнонауковий аналітичний метод, а також системний підхід. Для практичного вирішення поставлених задач використовувалися такі методи:

- загальнонауковий аналітичний метод
- методи теорії БД для формування інформаційно-логічної моделі предметної області та БД;
- методи алгоритмічного програмування

Практичне значення: комунальні підприємства можуть стати набагато ефективнішими в своїй роботі, як зрештою і будь-які, якщо стануть діджиталізованими. Розробка інформаційної системи для комунального

підприємства з надання авторемонтних послуг дозволяє їй бути значно зрозумілішим людям, ближчим до молодих поколінь, розширити ринок надання послуг та автоматизувати процес опрацювання замовлень чи навіть обслуговування клієнтів разом зі збором зворотньо зв'язку. В кінцевому рахунку виходить, що поява інформаційної ситсеми комунального підприємства, як приклад його діджиталізації, вирішує основні наявні зараз проблеми ффункціонування комунальних підприємств.



РОЗДІЛ 1. АНАЛІЗ ТЕОРЕТИКО-ПРАКТИЧНИХ ПІДХОДІВ ДО ВИРІШЕННЯ ЗАДАЧІ УПРАВЛІННЯ ПІДПРИЄМСТВОМ

1.1 Сучасний стан програмних рішень в області автоматизації управління підприємством

В даний час існують автоматизовані системи для підприємств, такі як: 1С, SLS - автосервіс, ZETA SERVIC, Універсальна Система Обліку 2.0 і багато інших [3].

До найбільш поширених автоматизованих систем можна віднести наступні:

- "1С";
- "SLS - автосервіс";
- "ZETA SERVIC".

Програма "1С" призначена для вирішення широкого спектра завдань автоматизації обліку та управління, що стоять перед сучасними підприємствами, які динамічно розвиваються [30].

До **переваг** даної програми можна віднести наступне:

- за допомогою "1С" можна вести всі існуючі види обліку робіт і матеріалів;
- на сьогоднішній день "1С" є однією з найбільш універсальних програм, яка може використовуватися в самих різних організаціях;
- програма "1С" має високу продуктивність, що дає можливість вирішувати з її допомогою найскладніші завдання;

Також "1С" володіє і рядом **недоліків**, до яких можна віднести наступне:

- в переважній більшості випадків, щоб "1С" вирішувала всі поставлені перед нею завдання, програму доводиться допрацьовувати; Кожне підприємство унікальне, тому для ефективної його роботи, як правило, потрібні індивідуальні рішення по автоматизації бізнес-процесів;

- при переході на "1С" з іншої програми можуть виникнути серйозні труднощі при перенесенні інформації з однієї бази даних в іншу;
- в "1С" утруднений пошук помилок, зроблених під час обробки документів;
- програма "1С" досить складна в освоєнні і вимагає спеціального навчання користувачів.

Програма *SLS-автосервіс* - це комплексне рішення для обліку в компаніях, що спеціалізуються на виконанні робіт по сервісному обслуговуванню різних транспортних засобів [22].

Програма має потужні інструменти, які дають можливість моделювати весь ланцюжок бізнес-процесів надання авторемонтних послуг - починаючи від надходження автомобіля на обслуговування, виконання ремонтних робіт і закінчуючи збором інформації для повного аналізу діяльності комунального підприємства з надання авторемонтних послуг і подальшого використання результатів.

SLS-Автосервіс дозволяє:

- працювати з замовлення-нарядами;
- роздруковувати необхідні документи;
- вести облік транспортних засобів, що знаходяться в обслуговуванні;
- враховувати запчастини та витратні матеріали;
- враховувати виконані роботи;
- проводити взаєморозрахунки з власниками автотранспорту;
- здійснювати нарахування співробітникам компанії за виконані роботи;
- вести фінансовий облік діяльності комунального підприємства з надання авторемонтних послуг.

У програму вбудовано докладний опис всіх функцій і можливостей. SLS-Автосервіс розрахований як на одного користувача так і мережевому варіанті.

Система *Zeta Service* - це програма обліку для комунального підприємства з надання авторемонтних послуг, розроблена на платформі 1С: Підприємство [22]. Вона дозволяє автоматизувати ведення оперативного, управлінського, складського і фінансового обліку. Одною з важливих **переваг** продукту є зручний інтерфейс, який налаштовується для кожного співробітника окремо.

Інтерфейс програми досить продуманий. Великі кнопки, які видно тільки тоді, коли потрібно, але це тільки після спеціалізованого налаштування кожного робочого місця. Що також вимагає або кваліфікованого фахівця запрошеного з боку підприємства обслуговуючої програми 1С, або мати штатного співробітника, навченого підприємством, що обслуговує програми сімейства 1С, що також коштує не малих грошей.

Решта представників програмного забезпечення в даному сегменті створені приблизно на тих же умовах що і попередні, це або програми сімейства 1С, або програми створені іншими розробниками з тими чи іншими вадами, які виправляти крім програмістів саме цієї організації практично нікому.

Таким чином, рішення, які існують на даний момент, є універсальними. Однак вони платні, складні в освоєнні і досить громіздкі для даного підприємства. Також, в деяких багатьох зайвих функцій, в інших - ні функцій, необхідних для комунального підприємства з надання авторемонтних послуг. Тому було прийнято **рішення** про необхідність власної розробки, яка буде відповідати всім поставленим вимогам і мати інтуїтивно зрозумілий інтерфейс. І не зажадає систематичної покупки ліцензій, що є важливим фактом. Програмний продукт, створений в рамках дипломного проекту, перш за все орієнтований на користувачів, які не мають спеціалізованих комп'ютерних навичок, а саме для працівників **комунального підприємства**

1.2. Характеристика програмних засобів та засобів створення та ведення баз даних

Під час аналізу вимог до системи основна увага приділялася з'ясуванню того, що повинно бути зроблено, незалежно від того, як це зробити. На етапі розробки системи вирішується питання, як реалізувати рішення, прийняті на етапі аналізу [31].

Спочатку розробляється загальна структура (архітектура) системи. Архітектура системи визначає її розбиття на модулі, задає контекст, в рамках якого приймаються проектні рішення на наступних етапах розробки [6].

Для розробки архітектури необхідно вибрати систему управління базами даних (СКБД). За способом доступу до баз даних (БД) розрізняють наступні СУБД [7]:

- клієнт-серверні;
- файл-серверні;
- вбудовані.

Клієнт-серверна система характеризується наявністю двох взаємодіючих самостійних процесів - клієнта і сервера, які можуть виконуватися на різних комп'ютерах, обмінюючись даними по мережі. За такою схемою можуть бути побудовані системи обробки даних на основі СУБД, поштові та інші системи.

У файл-серверної системи дані зберігаються на файловому сервері (наприклад, Novell NetWare або Windows NT Server), а їх обробка здійснюється на робочих станціях, на яких, як правило, функціонує одна з, так званих, "настільних СУБД" - Access, FoxPro, Paradox і тому подібні.

Додаток на робочій станції "відповідає за все" - за формування призначеного для користувача інтерфейсу, логічну обробку даних і за безпосереднє маніпулювання даними. Файловий сервер надає послуги тільки найнижчого рівня - відкриття, закриття та модифікацію **файлів**, не бази даних. База даних існує тільки в "мозку" робочої станції.

Безпосереднім маніпулюванням даними займається кілька незалежних і неузгоджених між собою процесів. Крім того, для здійснення будь-якої обробки всі дані необхідно передати через мережу з сервера на робочу станцію.

У клієнт-серверної системи функціонують (як мінімум) два додатки - клієнт і сервер, що ділять між собою ті функції, які в файл-серверній архітектурі виконує додаток на робочій станції. Зберіганням і безпосереднім маніпулюванням даними займається сервер баз даних, в якості якого може виступати Microsoft SQL Server, Oracle, Sybase, Firebird, Interbase, IBM DB2, Informix, PostgreSQL, MySQL, Caché, Лінтера [9].

Вибір системи управління баз даних (СУБД) є важким завданням і є одним з важливих етапів при розробці додатків баз даних. Обраний програмний продукт повинен задовольняти як поточні, так і майбутні потреби підприємства, при цьому слід враховувати фінансові витрати. Інформаційна система повинна відповідати найбільш загальним технічним вимогам [2].

Таблиця 1.1 - Характеристики інформаційної системи

Характеристики	Коментарі
Єдина база даних забезпечує розраховану на багато користувачів роботу.	Рекомендується використання централізованої бази даних на основі повноцінних промислових СУБД (MS SQL Server, Oracle, Informix, DB2).
Відсутність обмежень за кількістю об'єктів (максимальна кількість вимірювань, записів, звітів, число одночасно працюючих користувачів і т.д.).	Будь-які обмеження щодо кількості об'єктів, якими оперує система, говорять про недосконалість технічного рішення такого програмного продукту. Навіть якщо ці обмеження складають десятки тисяч одиниць і сьогодні здаються прийнятними, це означає, що в майбутньому виникне нездоланий бар'єр при

	розвитку бюджетної моделі.
Інтеграція з суміжними автоматизованими системами.	Можливості системи повинні дозволяти здійснювати повноцінний імпорт / експорт, при необхідності попередню обробку даних з різних облікових систем; бажана підтримка двостороннього зв'язку з наявними обліковими системами. Використання стандартних СУБД також полегшує інтеграцію.
Можливості доопрацювання системи на вимогу замовника.	Необхідно з'ясувати чи має постачальник програмного забезпечення можливість його доопрацювання на рівні програмного коду.

Розглянемо більш докладно найбільш поширені СУБД.

1) СУБД *Oracle* одна з найбільш потужних сучасних СУБД, призначених для реалізації баз даних рівня корпорації, що висуває серйозні вимоги до сервера. Oracle може працювати в більшості операційних систем: Windows-NT, 2000, Linux, UNIX, AIX, Nowell Netware [38].

Використання Oracle в якості СУБД дає можливість вибору мови програмування. Традиційно для цього використовується мова PL / SQL, але можна використовувати і набагато потужнішу мову програмування Java.

Oracle має в своєму розпорядженні потужні і зручні засоби адміністрування не тільки одного сервера, але і групи серверів, розташованих в різних частинах планети.

Основними перевагами Oracle можна вважати підтримку баз даних дуже великого обсягу (до 64 Гбайт), потужні засоби розробки і адміністрування, підтримку багатопроцесорності і двох мовних середовищ, а також інтеграцію з Web. Разом з цим програма має серйозні апаратні вимоги та високу ціну.

2) СУБД *Borland Interbase* містить все, що потрібно від СУБД, призначеної для потреб малого та середнього бізнесу [10]. До того ж починаючи з версії 6.0 програма стала безкоштовною, що істотно впливає на її використання різними підприємствами. Програма не вимоглива до апаратної частини. *Borland Interbase* підтримується платформами *Windows* і *Linux*, а також *UNIX*, *NetBSD*, *FreeBSD*.

Популярні мови програмування від *Borland*, як *Delphi*, *Kylix* і *C++ Builder*, надаються з компонентами, що дозволяють працювати з даною СУБД. Саме це дозволяє досягти дуже високої швидкодії програми.

3) СУБД *MySQL* набула широкого поширення в якості засобу роботи з базами даних в Інтернеті. Програма абсолютно невимоглива до ресурсів сервера, на якому працює, дуже швидка і до того ж абсолютно безкоштовна: вихідні коди для різних платформ доступні на сайті в Інтернеті. Спочатку програма була орієнтована на операційну систему *Linux*, але зараз вже існують версії програми для операційних систем *Windows*, *UNIX*, *NetBSD*, *FreeBSD*, *AIX*. Останнім часом програма завойовує популярність у користувачів *Macintosh* з використанням операційної системи *Mac OSX* [4].

5) СУБД *MS Access* використовується для вирішення локальних офісних завдань з обмеженим обсягом даних і формування звітів по результатам роботи, при цьому звіти можуть бути представлені в стандартному для офісних додатків вигляді.

MS Access одночасно є і середовищем розробки на двох мовах програмування (*Visual Basic* і сильно урізаний діалект *SQL*), і CASE-засобом, а також потужним і наочним засобом створення звітів за результатами роботи [20].

Програмне забезпечення дозволяє створювати програми, що складаються з одного файлу, що містить як текст програми, так і реляційну базу даних складної структури. *Access* легко інтегрується з іншими рішеннями від

Microsoft. Це дозволяє використовувати її як клієнтську частину інформаційного комплексу в зв'язці з MS SQL Server, яка виступає в якості серверної частини.

б) *MS SQL Server*. Протягом багатьох років продукти MS SQL Server характеризуються надійністю, безпекою, високою продуктивністю і зручністю в роботі. Сучасна СУБД MS SQL Server це найпотужніший програмний комплекс, що дозволяє створювати додатки будь-якої складності. Ядром цього комплексу є база даних, що зберігає інформацію, кількість якої за рахунок коштів, що надаються, масштабуються практично безмежно. З високою ефективністю працювати з цією інформацією одночасно може будь-яка кількість користувачів, не проявляючи тенденції до зниження продуктивності системи при різкому збільшенні їх числа [41].

Механізми масштабування в СУБД MS SQL Server останніх версій дозволяють збільшувати потужність і швидкість роботи сервера MS SQL Server і своїх додатків.

Ще однією складовою успіху СУБД MS SQL Server є те, що вона поставляється практично для всіх існуючих на сьогодні операційних систем. Таким чином, компаніям, які розпочинають роботу з продуктами MS SQL Server не доводиться змінювати вже складене мережеве оточення. Існує лише невелика кількість відмінностей при роботі з СУБД, обумовлених особливостями тієї чи іншої операційної системи. В цілому ж це завжди та ж сама безпечна, надійна і зручна СУБД MS SQL Server.

Потрібно відзначити мудру міграційну політику MS SQL Server. Розуміючи, що перехід з більш старої версії СУБД на нову, досить трудомістка процедура. Пов'язана вона з тестуванням роботи існуючих додатків в новому оточенні. MS SQL Server, при випуску нових продуктів приділяє особливу увагу сумісності знизу-вгору, роблячи цей перехід практично «безболісним». Останні версії СУБД MS SQL Server значно простіші в установці і початковому

налаштуванні. Також зросли можливості по спеціалізованому налаштуванні роботи СУБД під конкретну задачу [9].

На вибір серверу СУБД MS SQL Server вплинуло декілька факторів.

По-перше, технічні характеристики, які повністю задовольняли поставлену вимогу, а по-друге, доступність даної СУБД. Серед всіх переваг MS SQL Server слід виділити наступні:

- простота і зручність адміністрування;
- невибагливість і мінімальні системні вимоги;
- ефективність і швидкодія;
- високий ступінь інтеграції в середовище розробки;
- висока надійність;
- можливість розширення бази даних;
- наявність універсальних засобів захисту інформації;
- орієнтованість на Інтернет-технології;
- порівняно низька ціна.

Виходячи з перерахованих вище переваг, MS SQL Server було обрано як оптимальне рішення поставленого завдання, для реалізації СУБД [1].

РОЗДІЛ 2. ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ УПРАВЛІННЯ ДІЯЛЬНІСТЮ КОМУНАЛЬНОГО ПІДПРИЄМСТВА

2.1 Постановка задачі на розробку ІС

Для комплексної автоматизації всієї роботи комунального підприємства з надання авторемонтних послуг необхідно розібрати основні режими його роботи. Але, загалом, завдання по автоматизації роботи автосервісу повинна дозволяти вирішувати такі питання:

- використовувати весь спектр необхідних первинних документів для оформлення послуг по ремонту (заявка на ремонт, наряд-замовлення, талон на обслуговування);
- вести список автомобілів для будь-якого клієнта фірми, для кожного автомобіля зберігати набір необхідних реквізитів: державний номер, модель, технічні характеристики і т. п.
- отримувати історію обслуговування для кожного автомобіля;
- забезпечувати узгоджену роботу декількох користувачів при оформленні ремонтних документів;
- контролювати незавершений ремонт;
- фіксувати зриви робіт через скарг від клієнтів;
- вести облік запчастин;
- оформляти запчастини, надані для ремонту клієнтом;
- оформляти поставки запчастин;
- враховувати недопоставки, а також проводити інвентаризацію;
- оформляти замовлення від клієнтів на придбання запчастин, а також замовлення постачальникам на поставку товарів [16].

Сучасне життя досить відрізняється від того, яким жили люди ще років 50 тому. Сьогодні автомобіль - наш супутник життя і уявити себе без машини -

практично неможливо. День у день ми стикаємося з автомобілями, і, як наслідок, з їх ремонтом.

Ремонту автомобілів піддається практично кожен авто-власник. В умовах сучасного ринку займатися ремонтом автомобіля самостійно стало незручно і не вигідно. У авто-власника просто не залишається часу, щоб самостійно розібратися в своєму автомобілі. А з розвитком ринку іноземного автомобіля (ускладненням конструкції самого автомобіля) це стало і неможливо. Сучасний автомобіль вимагає турбот досвідченого, добре підготовленого механіка. З появою авто-сервісів на ринку послуг ремонт автомобілів став здійснюватись набагато простіше - цим стали займатися обізнані й досвідчені фахівці, які мають можливість без праці впоратися з будь-якою з проблем.

Зараз ремонт автомобілів здійснюється з використанням сучасного діагностичного і технологічного обладнання, тому що воно є засобом для отримання якісного результату.

Віддаючи свій автомобіль до комунального підприємства з надання авторемонтних послуг, будь-який з авто-власників повинен бути впевнений, що всі роботи по ремонту автомобіля будуть проведені на високому професійному рівні і з гарантією. Більшість сучасних автомобілів, навіть для профілактичних заходів вимагають специфічного обслуговування. Комунальне підприємство з надання авторемонтних послуг повинні справлятися з будь-яким із поставлених завдань. Авто-власнику досить пригнати свій автомобіль до комунального підприємства з надання авторемонтних послуг, і в короткі терміни всі проблеми з ним будуть вирішені.

Комунальне підприємство з надання авторемонтних послуг має штат висококваліфікованих фахівців, які пройшли відповідне навчання, мають достатній досвід роботи, і мають бажання надійно і якісно виконувати свою роботу. Комунальне підприємство з надання авторемонтних послуг може бути фактично- це весь спектр послуг по обслуговуванню автомобіля, який може

включати в себе не тільки ремонтні, кузовні та слюсарні роботи, фарбування і полірування автомобіля, але і буксирування, і транспортування вашого авто, і навіть надання юридичних послуг [15]. Наявність інформаційної системи, яка виконує більшість рутинної роботи, дозволить значно знизити витрати на виконання даних робіт. Це дозволить знизити собівартість послуг, що надаються.

Також автоматизована, злагоджена робота всієї команди співробітників в Комунальне підприємство з надання авторемонтних послуг, яка цінує не тільки свій час, але і, в першу чергу, час клієнта, буде гарантовано виробляти всі роботи, піклуватися про свого клієнта і про автомобіль цього клієнта. Завдяки наявності автоматизованої системи в Комунальне підприємство з надання авторемонтних послуг співробітники зможуть нагадувати клієнтам про проходження планового огляду технічного стану його автомобіля (ця послуга може надаватися клієнтам з їх згоди).

Користувачами даного програмного продукту будуть менеджери, які укладають договори з клієнтами, робітники, які виконують весь спектр робіт, зазначених в документі, отриманому після закінчення роботи з клієнтом по виявленню несправності, касир, який здійснює розрахунок з клієнтом також за отриманими в результаті розрахунку документами.

Розроблена програма призначена також і для використання її бухгалтером, директором і адміністратором на Комунальне підприємство з надання авторемонтних послуг, що займається великою кількістю всіляких операцій, пов'язаних з оглядом автомобіля, веденням та обліком ходу виконання робіт, контролем робітників і рухом грошових мас [19].

Таким чином, в даному сегменті ринку існує потреба в інформаційній системі, яка забезпечувала б автоматизовану обробку і маніпулювання даними, облік взаєморозрахунків та в будь-який момент надавала б оперативну інформацію. Дані в цій системі повинні бути згруповані по клієнтам, за типами

договорів, за списками автомобілів і т. д. - з метою надання повної інформації для аналізу і використання в розвитку авто-сервісного бізнесу.

При розробці такої системи необхідно приділити увагу наступним вимогам [32]:

- Простота використання. Системою повинен вміти користуватися оператор без спеціальної комп'ютерної підготовки, фахівець у своїй предметній області.
- Ефективність. Система повинна бути оснащена потужною і добре продуманою системою навігації і пошуку інформації.
- Надійність. В якості платформи потрібно використовувати інструментальні програмні засоби професійного рівня.
- Масштабованість. Розроблена система повинна легко розширюватися і забезпечувати гнучкі можливості для настройки.
- Виконання програми у відповідності зі стандартами побудови сучасних інформаційних систем. Вона повинна мати інтуїтивно-зрозумілий дружній інтерфейс.
- Доступність системи за вартістю для потенційного покупця. По можливості, ціна проекту повинна бути нижче, ніж у існуючих і потенційних аналогів.
- Відповідність системи сучасним вимогам по експлуатаційним параметрам. Система повинна бути простою в обслуговуванні та освоєнні користувачами.
- Високий рівень безпеки і захищеності інформації.
- Можливості розвитку в майбутньому.

2.2 Проектування логічної моделі даних

Метою бази даних, яка розробляється для користувачів, є облік клієнтів, співробітників, постачальників, та робіт, виконаних співробітниками. Користувачами бази даних «автосервіс» є працівники комунального підприємства з надання авторемонтних послуг.

Реляційна БД - основний тип сучасних баз даних. Складається з таблиць, між якими можуть існувати зв'язки з ключових значень [38].

Таблиця бази даних (table) - регулярна структура, яка складається з однотипних рядків (записів, records), розбитих на стовпці (поля, fields).

В теорії реляційних баз даних синонім таблиці - відношення (relation), в якому рядок називається кортежем, а стовпець називається атрибутом.

У концептуальній моделі реляційної БД аналогом таблиці є сутність (entity), з певним набором властивостей - атрибутів, здатних приймати певні значення (набір допустимих значень - домен).

Ключовий елемент таблиці (ключ, regular key) - таке її поле (простий ключ) або рядковий вираз, утворений із значень декількох полів (складових ключа), за яким можна визначити значення інших полів для однієї або декількох записів таблиці. На практиці для використання ключів створюються індекси - службова інформація, що містить впорядковані відомості про ключові значення. У реляційній теорії і концептуальній моделі поняття "ключ" застосовується для атрибутів відносин або сутності.

Первинний ключ (primary key) - головний ключовий елемент, однозначно ідентифікує рядок в таблиці. Можуть також існувати альтернативний (candidate key) і унікальний (unique key) ключі, слугує також для ідентифікації рядків в таблиці.

У реляційній теорії первинний ключ - мінімальний набір атрибутів, однозначно ідентифікує кортеж у відношенні.

У концептуальній моделі первинний ключ - мінімальний набір атрибутів сутності, однозначно ідентифікує екземпляр сутності [39].

Зв'язок (relation) - функціональна залежність між об'єктами. У реляційних базах даних між таблицями встановлюються зв'язки по ключах, один з яких в головній (parent, батьківської) таблиці - первинний, другий – зовнішній ключ - у зовнішній (child, дочірньої) таблиці, як правило, первинним не є і утворює зв'язок "один до багатьох" (1: N). У разі первинного зовнішнього ключа зв'язок між таблицями має тип "один до одного" (1: 1). Інформація про зв'язки зберігається в базі даних.

Зовнішній ключ (foreign key) - така підмножина атрибутів дочірнього відношення, що для будь-якого його непорожнього значення обов'язково знайдеться рівне значення первинного ключа головного відношення.

Реляційна модель:

- Клієнт (код клієнта, найменування, контакти)
- Автомобіль (код авто, марка, модель, реєстраційний номер)
- Виконавець замовлення (код замовлення, код співробітника, відсоток участі)
 - Зовнішні ключі: код співробітника, посилається на таблицю «співробітник».
- Співробітник (код співробітника, ПІБ, код посади, контакти)
 - Зовнішні ключі: код посади, посилається на таблицю «посаду».
- Робота (код роботи, найменування, код одиниці виміру, контакти)
 - Зовнішні ключі: код одиниці виміру, посилається на таблицю «одиниці виміру».
- Замовлення (код замовлення, дата, код клієнта, код авто, причина, стан)
 - Зовнішні ключі: код клієнта, код авто, посилаються на таблиці «клієнт», «автомобіль».

- Посада (код посади, найменування, оклад)
- Повноваження (код посади, об'єкт доступу, читання, зміна, видалення)
- Одиниці виміру (код одиниці виміру, найменування)
- Запаси (код запасів, номер за каталогом, найменування, виробник, код одиниці виміру, ціна відпускна)
- Зовнішні ключі: код одиниці виміру, посилається на таблицю «одиниці виміру»
- Постачальники (код постачальника, найменування, реквізити, контакти)
- Місце зберігання (код місця, найменування)

Модель даних, зроблена в MS SQL Server, представлена на рис. 2.1

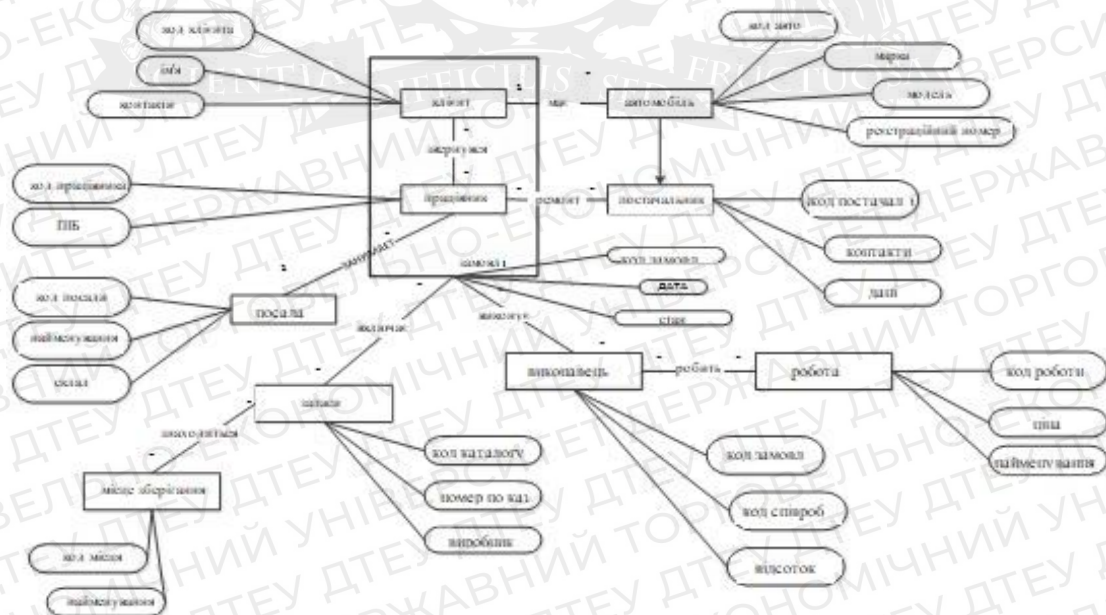


Рис. 2.1. Модель БД

2.3 Проектування фізичної моделі даних ІС

На основі концептуальної і реляційної моделей були спроектовані таблиці БД в MS SQL.

На рис. 2.2 показана таблиця з даними про клієнтів. Для того, щоб оформити замовлення, адміністратору для початку необхідно буде занести контактні дані клієнта в цей довідник, а вже в формі замовлення вибрати зі списку клієнта [42]. Такий спосіб дозволить уникнути втрати контактних даних клієнтів і помилок при введенні прізвища, імені та по батькові.

Посади	Повноваження	Працівники	Одиниці виміру	Місця зберігання	Запаси	Постачальник	Роботи	Клієнти
		КодКлієнта	Найменування	Контакти	Примітка			
		1	Сидоров Сергій Петрович	(063) 555 55 55				
		2	ТОВ Спецперевезення	(095) 125 55 89	знижка 10% на всі види робіт			

Рис. 2.2. Таблиця «клієнти»

На рис. 2.3 представлена таблиця з даними про співробітників, тут зберігаються логіни і паролі для входу в інформаційну систему, паролі зберігаються в зашифрованому вигляді, алгоритм хешування MD5.

Посади	Повноваження	Працівники	Одиниці вимірування	Місця зберігання	Запаси	Постачальник	Роботи	Клієнти	Авто
		Код	ПІБ	Посада	Контакти	Примітка	Ім'яКористувача	Пароль	
		3	Белешко Андрій Вікторович	Власник			owner	72 12 2c e9 6...	
		4	Овдєнко Володимир Павлович	Адміністратор			admin	21 23 2f 29 7a...	

Рис. 2.3. Таблиця «працівники»

На рис. 2.4 показана таблиця посад в комунальному підприємстві з надання авторемонтних послуг і оклад для кожної посади.

Посади	Повноваження	Працівники	Одиниці виміру	Місця зберігання	Запаси	Постачальник	Роботи	Клі
	КодПосади	Найменування	Оклад	Примітка				
▶	12	Власник	0					
	15	Адміністратор	30000					
	16	Майстер-приймщик	20000					
	17	Комірник	20000					
	18	Автослусар	10000		+ % від робіт			
	19	Автомаляр	10000		+ % від робіт			

Рис. 2.4. Таблиця «посади»

На рис. 2.5 показана таблиця повноважень для кожної посади, ці повноваження визначають доступ до інформаційної системи.

Посади	Повноваження	Працівники	Одиниці Виміру	Місця зберігання	Запаси	Постачальник	Роботи	Клієнт
Обрати посаду		Адміністратор						
Посада	Об'єкт доступу	Читання	Зміна	Видалення				
▶	Адміністратор	Результати	1	0	0			
	Адміністратор	ДовідникКлієнтів	1	1	0			
	Адміністратор	ДовідникАвто	1	1	0			
	Адміністратор	ДовідникПрацівників	1	1	0			
	Адміністратор	ДовідникПосад	0	0	0			
	Адміністратор	ДовідникМісцьЗберігання	1	1	1			
	Адміністратор	ДовідникОдиницьВимірювання	1	1	1			
	Адміністратор	ДовідникЗапасів	1	1	0			
	Адміністратор	ДовідникРобіт	1	1	0			
	Адміністратор	ДовідникПостачальників	1	1	0			
	Адміністратор	ЗамовленняОсновнаІнформація	1	1	0			
	Адміністратор	ЗамовленняТовари	1	1	1			
	Адміністратор	ЗамовленняРоботи	1	1	1			
	Адміністратор	ЗамовленняВиконавці	1	1	1			
	Адміністратор	ПрихідЗапасів	1	1	0			
	Адміністратор	МатрицяПовноважень	0	0	0			

Рис. 2.5. Таблиця «повноваження»

На рис. 2.6 представлена таблиця запасів, в ній перераховані всі запчастини, які використовуються в роботах. Даний перелік буде використовуватися при оформленні замовлення на поставку запчастин майстром-приймальником.

Посади	Повноваження	Працівники	Одиниці Виміру	Місця зберігання	Запаси	Постачальник	Роботи	Клієнти	Авто
	Код запасів	Номер по каталогу	Найменування	Виробник	Од. вим.	Ціна відпускна	Примітка		
	9313	21120-1164010-20	АДСОРБЕР ВА3-2110 АВТОВА3 ЕВ...	АВТОВА3	шт	515			
	9314	21103-1164010-02	АДСОРБЕР ВА3-2110 АВТОВА3 ЕВ...	АВТОВА3	шт	489			
	9315	21112-1164009-00	АДСОРБЕР ВА3-2110 АВТОВА3 С.О.	АВТОВА3	шт	505			
	9316	21214-1164009-00	АДСОРБЕР ВА3-21214 АВТОВА3	АВТОВА3	шт	777			

Рис. 2.6. Таблиця «запаси»

На рис. 2.7 показана таблиця місць зберігання. В комунальному підприємстві з надання авторемонтних послуг є два склади, це основний склад і склад паливно мастильних матеріалів. Даний перелік буде використовуватися майстром-приймальником при оформленні поставки.

Посади	Повноваження	Працівники	Одиниці вимірювання	Місця Зберігання	Запаси
	Код Місця Зберігання	Найменування	Примітка		
	1	Основний склад	Графік роботи Пн-Пт 8:00-...		
	2	Склад ГЗМ			
*					

Рис. 2.7. Таблиця «місця зберігання»

На рисунку 2.8 показана таблиця одиниць виміру.

	КодОдвим	Найменування
▶	1	кг
	2	л
	3	м2
	4	шт
	5	дец2
	6	компл
*		

Рис. 2.8. Таблиця «одиниці виміру»

На рисунку 2.9 представлена таблиця постачальників, в ній зберігаються дані про постачальників. Дані цієї таблиці потрібні для оформлення поставки майстром-приймальником.

Посади	Повноваження	Працівники	Одиниці виміру	Місця зберігання	Запаси	Постачальник	Роботи	Клієнти	Авто
			Найменування	Реквізити	Контакти	Примітка			
▶			Авто-Альянс	м. Харків, вул. Євгена Коновальця...	(066) 585 33 88	з 9:00 до 18:00. Сб і Нд - ви...			
			АВТОЛІГА	16658, м. Київ, проспект Юрія Гага...	(055) 555 55 55	Запчастини			
*									

Рис. 2.9. Таблиця «Постачальники»

Під час проектування бази даних так само потрібно подумати про цілісність даних. Правильна структура таблиць дозволяє захистити дані від порушення зв'язків і внесення невірних значень. Необхідно визначити найкращий шлях забезпечення цілісності даних. Цілісність даних ґрунтується на стійкості і точності даних, які зберігає база даних.

У таблицях БД була використана цілісність полів. У полях, таких як найменування, код замовлення, номер документа, неможливо використовувати нульові значення, так як це призведе до втрати необхідних даних про замовлення.

Для збереження даних, була дотримана цілісність таблиці. Всі рядки в таблицях мають унікальний ідентифікатор, названий *первинним ключем*. При видаленні будь-якої посади з БД, необхідно позначити її на видалення, щоб забезпечити збереження всіх даних, пов'язаних з видаленою посадою.

Так само дотримана цілісність посилань. Відносини між первинним ключем (таблиці, на яку посилаються) і зовнішнім ключем (таблиці, яка посилається на іншу) завжди захищені. Рядок основної таблиці, на яку посилаються, не може бути видалений і первинний ключ не може бути змінений, якщо вторинний ключ посилається на рядок, доки не буде знищений зв'язок. Інакше зв'язок порушується і відновити його потім стає проблематично.



РОЗДІЛ 3. РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ УПРАВЛІННЯ ДІЯЛЬНІСТЮ КОМУНАЛЬНОГО ПІДПРИЄМТВА

3.1. Загальна архітектура ІС

Розробка будь-якої програми включає в себе кілька етапів. Для початку необхідно визначитися - яка функціональність нам необхідна і, відповідно, яка структура програмного забезпечення зможе це реалізувати. Потім, нам необхідно вибрати спосіб взаємодії з даними (це основна функція нашої програми) і створити необхідні для цього інструменти [37]. На останньому етапі ми повинні забезпечити реалізацію призначеного для користувача інтерфейсу в нашій програмі і реалізувати допоміжні функції.

Згідно поставленого завдання, наша програма повинна:

- Надавати можливість авторизації різних користувачів системи;
- Надавати можливість перегляду таблиць та форм;
- Забезпечувати введення інформації у формах;
- Забезпечувати можливість редагування інформації в таблицях;
- Формувати остаточний звіт по зробленій роботі.

Система забезпечує зберігання даних про клієнтів, замовлення, автомобілі, запчастини, роботи і співробітників. Обробка інформації вручну займає багато часу, як при заповненні бланків, так і при пошуку і виправлення некоректних записів. Забезпечуючи можливість управлінському персоналу коригувати дані безпосередньо в режимі реального часу, багато проблем можуть бути зменшені або зовсім вирішені.

1) Загальні відомості

Повне найменування системи: Інформаційна система «Автосервіс»

Умовне позначення системи: ІС

Призначення і область застосування: програма призначена для створення, управління вмістом бази даних:

- база клієнтів, співробітників, постачальників;
- дані про запаси;
- дані про витрати і доходи;
- дані про роботи.

Програма надає інтерфейс для управління вмістом бази даних.

2) Вимоги до системи

Система забезпечує можливість переглянути опції:

- поділ користувачів, які підключаються через інтерфейс на групи в залежності від займаної посади, доступ до всіх даних повинен бути тільки у керівника комунального підприємства з надання авторемонтних послуг;
- можливість введення і редагування інформації в базі даних;
- наявність зрозумілого інтерфейсу для оформлення замовлення;
- можливість обліку запасів на складі.

3) Вимоги до надійності

- Здійснюється розмежування прав доступу до системи.
- Ведеться журнал подій системи.
- Відмови програми внаслідок некоректних дій користувача при взаємодії з програмою через інтерфейс неприпустимі.
- Кожен співробітник має доступ тільки до тих даних, які необхідні для його роботи. Генеральний директор має доступ до всіх даних інформаційної системи, але тільки для читання.
- Паролі та логіни зберігаються в довіднику співробітників, паролі в зашифрованому вигляді.

4) Вимоги до інформаційної та програмної сумісності та до програмних засобів, які використовуються програмою.

- Системні програмні засоби, що використовуються програмою, повинні бути представлені ліцензійною локалізованою версією операційної

системи Windows 7 або Windows 8. Наявність .NET Framework 4 і MS SQL Server CE Client.



3.2. Вхідні та вихідні форми ІС

Вхідними даними повинні бути:

- за заявками - номер, дата заявки, клієнт, вид послуг, що надаються, ;
- по клієнтах – ПІБ клієнта, контакти, урахування знижки;
- по автомобілях – код, марка, модель, VIN, регіональний номер;
- за видами робіт - найменування роботи, працівник, що виконується даний вид ремонту;

Вихідними даними повинні бути:

- сформований рахунок, за наданий ремонт;
- наряд-замовлення із деталями замовлення, ремонту.

3.3 Алгоритми роботи ІС

В процесі роботи над дипломною роботою була створена інформаційна система, яка забезпечує:

- підвищення ефективності управління;
- оптимізацію процесів збору, обробки, обліку та контролю інформації;
- підвищення якості обслуговування клієнтів, скорочення рутинної роботи;
- оперативність доступу до інформації для всіх підрозділів.

Система працює за принципом 1 автомобіль - 1 замовлення-наряд на рисунку 3.1 схематично показано як формується даний документ на комунального підприємства з надання авторемонтних послуг. Спочатку збираються дані про замовника та про автомобіль. Далі вибирається класифікація ремонту. Потім кожному класу ремонту підбираються відповідно роботи і деталі. Після всього цього список всіх робіт передається механіку або автослюсаря, а список необхідних деталей комірникові.

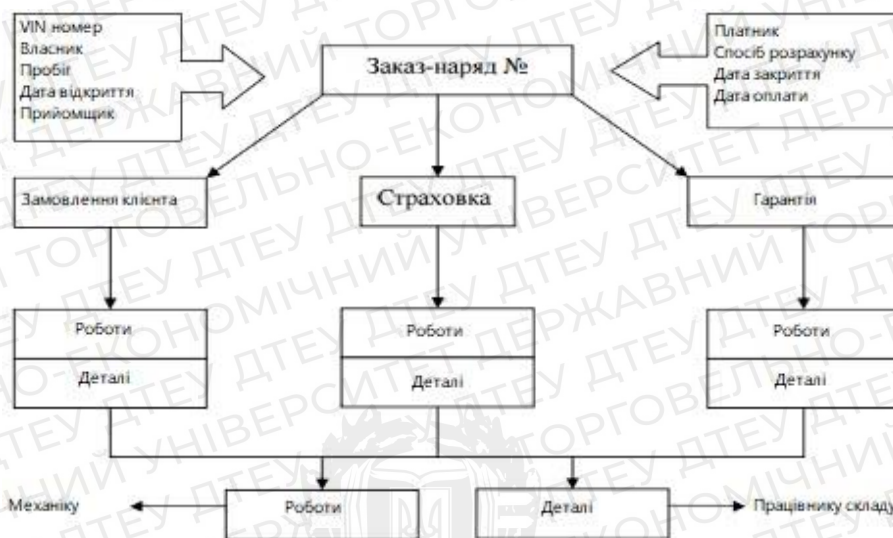


Рис. 3.1. Схема формування замовлення-наряду

3.4. Інструкція користувача

Для створення найбільш простого інтерфейсу і підвищення безпеки даних було прийнято рішення про розмежування доступу до бази даних. Логіни і паролі були створені адміністратором баз даних. Так як інформація, яка зберігається в базі даних, має фінансову звітність, було вирішено використовувати алгоритми шифрування для паролів [8]. При запуску програми з'являється вікно з пропозицією вибрати користувача.

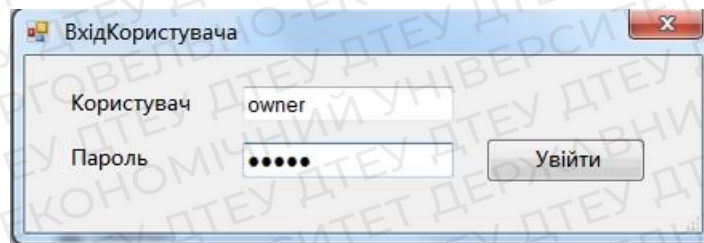


Рис. 3.2. Форма авторизації

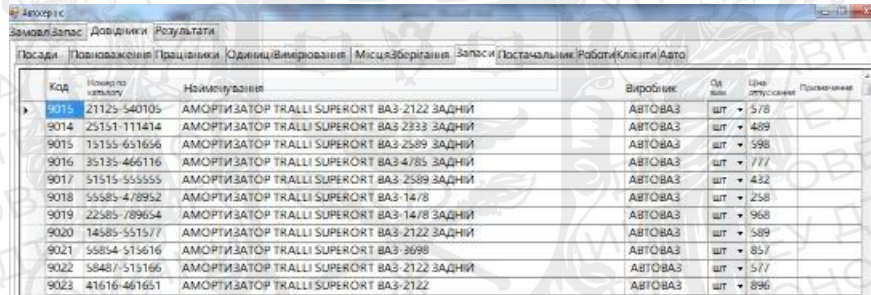
Необхідно вибрати співробітника і ввести пароль. Після натискання клавіші «Ок» від пароля обчислюється хеш-функція і отриманий результат порівнюється з даними в базі. Якщо авторизація проходить успішно,

завантажується робоча форма співробітника відповідно до його статусу. Подивитися код програми можна в Додатку.

Інтерфейс генерального директора включає в себе всю інформацію комунального підприємства з надання авторемонтних послуг, але доступна вона тільки для читання.

На даній формі є можливість для додавання і редагування виконуваних робіт і використовуваних в них запчастин. Все це здійснюється за допомогою додаткових форм.

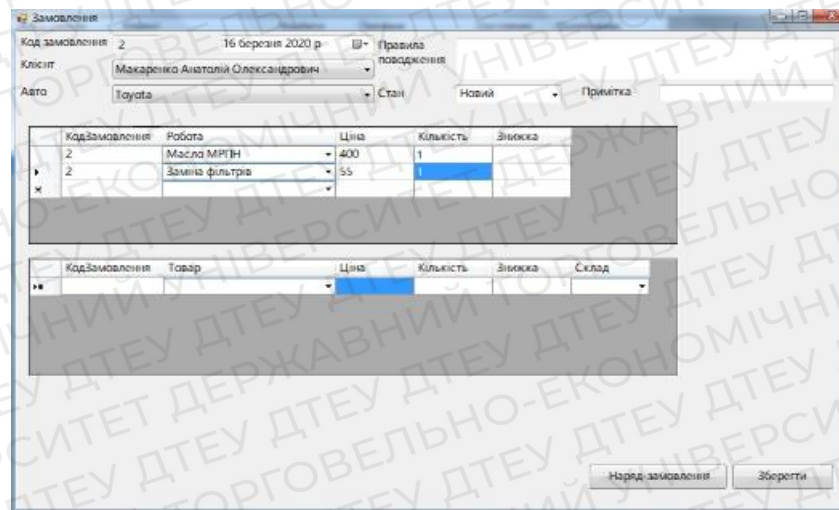
На вкладці «запаси» можна подивитися всі запчастини, масла, фарбу і інші матеріали, які є в наявності на складі.



Код	Номеро каталогу	Найменування	Виробник	Од. вим.	Ціна	Статус	Примітка
9019	21125-540105	АМОРИЗАТОР TRALLI SUPERORT ВА3-2122 ЗАДНИЙ	АВТОВА3	шт	578		
9014	25151-111474	АМОРИЗАТОР TRALLI SUPERORT ВА3-2333 ЗАДНИЙ	АВТОВА3	шт	489		
9015	15155-651656	АМОРИЗАТОР TRALLI SUPERORT ВА3-2589 ЗАДНИЙ	АВТОВА3	шт	598		
9016	35135-466116	АМОРИЗАТОР TRALLI SUPERORT ВА3-4785 ЗАДНИЙ	АВТОВА3	шт	777		
9017	51515-555555	АМОРИЗАТОР TRALLI SUPERORT ВА3-2589 ЗАДНИЙ	АВТОВА3	шт	432		
9018	55585-478952	АМОРИЗАТОР TRALLI SUPERORT ВА3-1478	АВТОВА3	шт	258		
9019	22585-789654	АМОРИЗАТОР TRALLI SUPERORT ВА3-1478 ЗАДНИЙ	АВТОВА3	шт	968		
9020	14585-551577	АМОРИЗАТОР TRALLI SUPERORT ВА3-2122 ЗАДНИЙ	АВТОВА3	шт	589		
9021	55854-515616	АМОРИЗАТОР TRALLI SUPERORT ВА3-3698	АВТОВА3	шт	857		
9022	58487-515166	АМОРИЗАТОР TRALLI SUPERORT ВА3-2122 ЗАДНИЙ	АВТОВА3	шт	577		
9023	41616-461651	АМОРИЗАТОР TRALLI SUPERORT ВА3-2122	АВТОВА3	шт	896		

Рис. 3.3. Інтерфейс генерального директора

В інтерфейсі адміністратора можливе додавання і зміна замовлень. Дані клієнта спочатку заносяться в довідник клієнтів, щоб уникнути втрату контактів, а потім вибирається зі списку в формі оформлення замовлення.



КодЗамовлення	Робота	Ціна	Кількість	Знижка
2	Масла МРГН	400	1	
2	Заміна фільтра	55	1	

КодЗамовлення	Товар	Ціна	Кількість	Знижка	Склад
---------------	-------	------	-----------	--------	-------

Рис. 3.4. Форма оформлення замовлення

Для того, щоб сформувати замовлення-наряд, потрібно просто натиснути кнопку «замовлення-наряд» і після оформлення замовлення з'явиться документ на друк у форматі html. На рисунку 3.5 показаний приклад замовлення-накладної.

Наряд-замовлення №2 від 16.03.2020

Клієнт: Макаренко Анастасій Олександрович

Авто: Toyota

Виконані роботи	Ціна	Кількість	Знижка	Сума
Масло МРПН	400,00	1	0 %	400,00
Заміна фільтрів	55,00	1	0 %	55,00
			Всього:	455,00

Запчастини	Ціна	Кількість	Знижка	Сума
			Всього:	0,00

В сумі за наряд-замовлення: 455,00

В умовах ремонту, стипендія та гарантія озброєні і згодні. Замовник: _____

Замовлення виконано _____ Майстер: _____

Претензій по якості і переліку робіт не маю, автомобіль прийняв. Замовник: _____

Рис. 3.5. Приклад замовлення-наряду

З головного меню можна зайти в управління довідниками системи. Натиснувши пункт меню «Довідники» відкривається форма. У довідниках знаходяться дані про співробітників, клієнтів, запаси, постачальників, роботи.

На рисунку 3.7 представлена форма для оформлення поставки, в неї вносить дані майстер-приймальник. Після збереження даних, в вкладку «залишки запасів» автоматично вносяться прийняті майстром-приймальником матеріали.

Система надає звіт по заданому періоду. (Рисунок 3.7, рисунок 3.8)

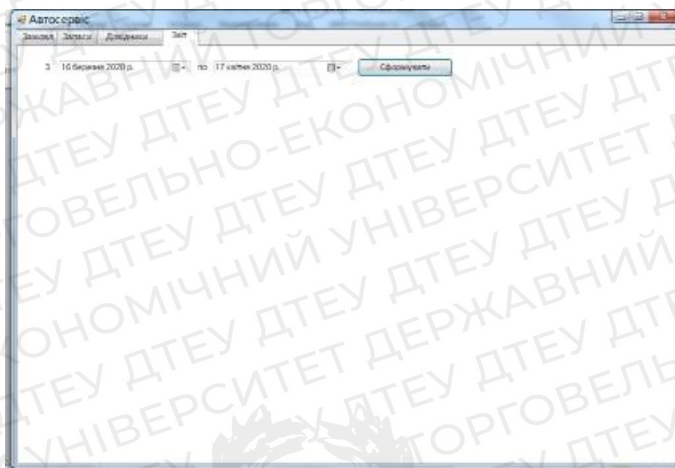


Рис. 3.7. Форма «звіт»

Звіт за період з 16.03.2020 по 17.04.2020

Замовлення

За період з 16.03.2020 по 17.04.2020 оформлено
замовлень: 7
на суму 3990.00 грн в т.ч.:
робіт на суму 160 грн;

Поставки

За період з 16.03.2020 по 17.04.2020 оформлено поставок
11
на суму 49 455.00 грн (кількість: 55).

Рис. 3.8. Звіт

Розроблена інформаційна система дозволяє співробітникам вести облік запасів на складі, облік клієнтів, стан робіт, облік співробітників, а так само проводити розрахунок витрат і доходів від виконаних робіт. В цьому розділі були відображені приклади роботи ІС, що функціонує завдяки програмному коду * ДОДАТОК: Програмний код реалізації додатку).

3.5 Характеристика задіяних засобів розробки програмного забезпечення та прикладних програм ІС

Для написання програми була обрано середовище програмування Visual Studio 2012 C#, засноване на мові програмування C#. Дане середовище вигідно відрізняється ефективністю і надійністю [11].

Для стабільного функціонування програми необхідний комп'ютер фірми IBM або сумісний з ним, з об'ємом оперативної пам'яті не менше 128 Мб., Процесор з частотою не менш 600МГц.

Для вирішення поставленого завдання необхідно використовувати функціональну, ефективну і зручну платформу для розробки, що дозволяє застосовувати принципи об'єктно-орієнтованого програмування. В якості такої платформи була обрано середовище .NET.

Середовище розробки Visual Studio, що поставляється разом з .NET, надає необхідний інструментарій для ефективного і швидкого створення додатків з графічним інтерфейсом.

Поява технології .NET спричинило масову реконструкцію деяких мов програмування, які прагнуть використовувати ті чи інші можливості платформи, такі як C++ і Visual Basic. Microsoft вирішили запропонувати розробникам альтернативу - мову, орієнтовану спеціально на .NET і створили C#. Самі розробники мови описують її, як просту, сучасну, об'єктно-орієнтовану і безпечну мову програмування. Синтаксично C # нагадує C ++ і Java, що дозволяє за досить короткий час вивчити тонкощі нової мови.

Незважаючи на те, що C # і .NET призначені в першу чергу для веб-розробки, їх також активно застосовують для створення додатків, які повинні встановлюватися на машині кінцевого користувача, де і буде виконуватися вся обробка даних. Розробку таких додатків забезпечує бібліотека Windows Forms, що дозволяє проектувати графічний інтерфейс. Система, описана в даній роботі, розроблена саме за допомогою бібліотеки Windows Forms [35].

Мова програмування C# претендує на справжню об'єктну орієнтованість.

Мова програмування C# покликана реалізувати компонентно-орієнтований підхід до програмування, який сприяє меншій машинно-архітектурній залежності результуючого програмного коду, більшої гнучкості, переносимості і легкості повторного використання програм [40].

Принципово важливою відмінністю від попередників є початкова орієнтація на безпеку коду.

Розширена підтримка подієво-орієнтованого програмування.

Мова програмування C# є «рідною» для створення додатків в середовищі Microsoft .NET, оскільки тісно і ефективно інтегрована з ним.

Microsoft Visual Studio- це версія Visual Studio і .NET Framework, яка підтримує нові та вдосконалені об'єкти, включає середовище розробки з оновленим інтерфейсом і відрізняється інтегрованою підтримкою Microsoft SQL Server, дозволяючи створювати і розгортати проекти із застосуванням сервера баз даних. [34].

Інтерфейс Visual Studio традиційно виконаний в одному стилі з MS Office. Є список завдань, в який поміщують інформацію про помилки та про необхідні доробки. Кожному пункту можна призначити пріоритет, а після виконання встановити прапорець, який повідомляє про завершення зазначеного завдання. Task List підтримує сортування записів по тексту, за пріоритетністю статусу. Властивості проекту в Visual Studio можна редагувати за допомогою вбудованого інструменту, який дозволяє змінювати налаштування і підписи збірки, посилання на зовнішні модулі, набір прав, необхідних для її функціонування. Крім того, розробник легко може зберегти налаштування свого призначеного для користувача IDE в файлі налаштувань і застосувати його в разі переходу на інший комп'ютер. Розробник баз даних може використовувати об'єктно-орієнтовані мови програмування, такі як C # і Visual Basic, спираючись на широкий спектр вбудованих можливостей класів і методів .NET Framework.

Крім того, програміст може скористатися компонентами, написаними сторонніми компаніями. З появою SQL Server був вдосконалений механізм доступу до даних.

Класи і структури є двома основними конструкціями системи загальних типів CTS в платформі .NET Framework. Кожна по суті є структурою даних, інкапсулює набір даних і поведінку. Дані і поведінку є членами класу або структури, і в них включені методи, властивості, події і так далі [33].

Оголошення класу або структури подібно кресленням, яке використовується для створення екземплярів або об'єктів під час виконання. При визначенні класу або структури з ім'ям Person, Person є ім'ям типу. При оголошенні або ініціалізації змінної р типу Person, р вважається об'єктом чи примірником Person. Можливе створення декількох екземплярів одного типу Person, і кожен екземпляр може мати різні значення в своїх властивостях і полях.

Клас є посилальним типом. При створенні об'єкта класу змінна, до якої призначається об'єкт, зберігає тільки посилання на пам'ять. При призначенні посилання на об'єкт до нової змінної нова змінна посилається на вихідний об'єкт. Зміни, внесені через одну змінну, відображаються в іншій змінній, оскільки обидві вони посилаються на одні дані.

Структура є типом значення. При створенні структури змінна, до якої вона призначається, зберігає фактичні дані структури. При призначенні структури нової змінної виконується її копіювання. Тому нова змінна і вихідна змінна містять дві окремих копії одних даних. Зміни, внесені в одну копію, не впливають на іншу копію.

Приклад класу Permission (Лістинг 1.1), використовується для визначення окремого набору прав для окремого користувача:

Лістинг 1.1.

```
public class Permissions: List <Permission>
{
    public Permission this [string name]
    {
        get
        {
            return this.FirstOrDefault (tTemp => tTemp.name == name);
        }
    }
}

public class Permission
{
    public string name;
    public bool canRead;
    public bool canEdit;
    public bool canDelete;
    public Permission (DataRow row)
    {
        name = row [ "Об'єктДоступу" ]. ToString ();
        canRead = (row [ "Читання" ]. ToString () == "1");
        canEdit = (row [ "Зміна" ]. ToString () == "1");
        canDelete = (row [ "Видалення" ]. ToString () == "1");
    }
}
}
```


ВИСНОВОК

В ході роботи була дана характеристика діючої системи керування та характеристика існуючої системи аналізу виконання робіт з системи ведення документів. Було запропоновано та розроблено нову автоматизовану систему аналізу виконання робіт з системи ведення документів.

З метою обґрунтування доцільності даної розробки були проведені дослідження та аналіз існуючих теоретико-практичних розробок у даному напрямку. Це дало змогу підтвердити, що розробка автоматизованої системи є доцільною і необхідною в умовах ринкової конкуренції. Вона дасть змогу підприємству вийти на новий рівень роботи з замовленнями, більш прогресивний та динамічний.

Сферою застосування спроектованої системи аналізу є різні відділи на комунальному підприємстві, зокрема і що було продемонстровано - Комунальне підприємство з надання авторемонтних послуг. За допомогою даної системи можна не тільки відслідкувати кількість виконаних і замовлених послуг, але й повністю проаналізувати роботу підприємства:

- Отримати звіт по тижнях
- Зробити аналіз виконаних послуг
- Проаналізувати темп росту виконаних замовлень

Даний програмний продукт зручний і простий у використанні. Користувачам буде досить легко освоїтись у інтерфейсі програми, а самому Комунальному підприємству з надання авторемонтних послуг не потрібні нові співробітники, а процес взаємодії з клієнтами стає набагато приємнішим, а тому і його ефективність та рентабельність буде значно виростати.

Надалі можливі поліпшення й доробки програмного продукту.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Адміністрування Microsoft SQL Server 2000. Навчальний курс MCSA / MCSE, MCDBA
2. А. П. Сиротинська, І. Д. Лазаришина. Інформаційні системи підприємств малого бізнесу, 2- 008. – 155 с.
3. Гайдамакин Н. А., Автоматизовані інформаційні системи, бази і банки даних. Вступний курс: Навчальний посібник. - М .: Геліос АРВ, 2002. - 368 с.
4. Кузнецов М.В. MySQL 5 / М.В. Кузнецов, И.В. Симдянов. – БХВ-Петербург, 2010. – 1024 с.
5. Сайт з SQL і клієнт / серверної технології [Електронний ресурс]. - Режим доступу: <http://www.sql.ru/>, вільний.
6. Петров В. М., Інформаційні системи, Підручник 2 изд., (Серія «Навчальний посібник») вид. ПІТЕР, 2006р., 656 с.
7. Галіцина О.А. Бази даних: навчальний посібник / О. А. Галіцина. - 4-е вид., Перероб. і доп. - Юрайт, - 2014. - 78с .;
8. Кларк, Д. Об'єктно-орієнтоване програмування в VisualBasic .NET. Бібліотека програміста / Д. Кларк. - СПб .: Пітер; - 2003.
9. Бекаревич Ю.Б., Пушкіна Н.В., Смирнова О.Ю. Управління базами даних. [Текст] / Бекаревич Ю.Б., Пушкіна Н.В. – Санкт-Петербург, 2009. - 754с.
10. Боррі Х. Firebird: керівництво розробника баз даних. В оригіналі. 2-е видання. СПб.: БХВ-Петербург, - 2007.
11. Джон Шарп. Microsoft Visual C #. Детальний опис, 8-е вид., 2017. – 655 с.
12. [Мартін Р., Мартін М.](#) Принципи, патерни і методики гнучкої розробки на мові C #, - 2011.
13. [Віссер Дж.](#) Розробка обслуговуючих програм на мові C #. Вид. [ДМК Пресс](#), - 2016.

14. Марков О. Д. Комунальні підприємства. Львів: Оріяна Нова, 1998. – 332 с.
15. Лудченко О. А. Технічна експлуатація і обслуговування автомобілів. Технологія – К.: Знання, 2009. – 478 с.
16. Канарчук В. Є., Лудченко О. А., Чигринець А. Д. Основи технічного обслуговування і ремонту автомобілів – К.: Вища шк., 2008. – 342 с.
17. Ким С. О., Пушкін П. С., Овчинников С. І. Організація і планування промислового виробництва, - Мінськ: «Вища школа», - 1980.
18. Кожекін Г. Я., Синиця Л. М. Управління Комунальним підприємством, - Мінськ: ПП «Екоперспектива», - 2008.
19. Організація, планування і управління діяльністю комунальних підприємств, - під ред С. Є. Каменіцера, Ф. Ф. Русинова, - Москва: «Вища школа», - 2009.
20. Ю. Бекаревич, Н. Пушкіна. Самовчитель MS Office Access 2016, - 2017.
21. Анісімов А. П. Організація і планування комунальних підприємств. – М.: Транспорт, - 1982.
22. Будрій А. Г., Коновалов Г. А. Економіка автомобільного транспорту. – Київ: видавничий центр «Академія», - 2012.
23. Семенов Г. А., Станчевський В. К. Організація і планування на підприємстві. – Київ «Центр учбової літератури», - 2006.
24. Польшиков В. І., Сахно Є. Ю. Економіка, організація та управління технічним обслуговуванням діяльності комунального підприємства. – Київ «Центр навчальної літератури», - 2010.
25. НДІАТ Короткий автомобільний довідник.
26. Норми витрат на матеріали і запасні частини.
27. Онищенко В. О., Старовірець А. С., Редкін О. В., Чевганова В. Я. Організація виробництва.

28. Польшаков В. І., Сахно Є. Ю. Економіка, організація та управління комунальним підприємством.
29. Роїна О. М. Автомобільний транспорт в Україні.
30. <http://lib.mdpu.org.ua/e-book/vstup/L5.htm> [Електронний ресурс].
31. Ризун Н. О. Сучасні інструменти аналізу складних економічних систем / Н. О. Ризун, Є. Г. Холод // Академічних огляд. – 2008. - № 1.
32. Трубілін І. Т. Автоматизовані інформаційні технології в економіці, 1999. – 416 с.
33. Ендрю Троелсен, Філіпп Джексіс. Мова програмування C # 7 і платформи .NET і .NET Core. Том 2, 2020.
34. Дж. Ріхтер. CLR via C #. Програмування на платформі Microsoft.NET Framework 4.5 мовою C #, 2016.
35. Брайан Нойс. Прив'язка даних у Windows Forms, 2009.
36. Таха Х. Введення у дослідження операцій. Пер. з англ. [Текст, електронний ресурс] / Хемді Таха. – М.: Вільямс, 2005.
37. Томашевський В. Моделювання систем [Текст] / В. М. Томашевський. – К.: БПУ, 2005.
38. Фролов А. Бази даних в Інтернеті [Текст] / А. В. Фролов, Г. В. Фролов. – М.: Російська редакція, 2000. – 448 с.
39. Чекалов А. Бази даних: від проектування до розробки додатків [Текст, електронний ресурс] / А. Чекалов – СПб.: БХВ-Петербург, 2003. – 384 с.
40. Янг Б. Об'єктно-орієнтовний аналіз і проектування с прикладами додатків. 3-є видання [Текст, електронний ресурс] / Боббі Янг, Джим Коннален, Граді Буч. – М.: Вільямс, 2008. – 720 с.
41. <http://lib.mdpu.org.ua/e-book/vstup/L5.htm> [Електронний ресурс].
42. Проектування і реалізація БД MS SQL Server 2000. Навчальний курс MCSA / MCSE, MCDBA.- Москва: Російська Редакція 2003, видання 2-е, виправлене.

ДОДАТОК

Програмний код реалізації додатку

```
///Represents the strongly named DataTable class.
```

```
///</summary>
```

```
[global::System.Serializable()]
```

```
[global::System.Xml.Serialization.XmlSchemaProviderAttribute("GetTypedTableSchema")]
```

```
public partial class КлієнтиDataTable :
```

```
global::System.Data.TypedTableBase<КлієнтиRow> {
```

```
private global::System.Data.DataColumn columnКодКлієнта;
```

```
private global::System.Data.DataColumn columnНайменування;
```

```
private global::System.Data.DataColumn columnКонтакти;
```

```
private global::System.Data.DataColumn columnПримітка;
```

```
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
```

```
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]
```

```
public КлієнтиDataTable() {
```

```
    this.TableName = "Клієнти";
```

```
    this.BeginInit();
```

```
    this.InitClass();
```

```
    this.EndInit();
```

```
}
```

```
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
```

```
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]
```

```
internal КлієнтиDataTable(global::System.Data.DataTable table) {
```

```

this.TableName = table.TableName;
if ((table.CaseSensitive != table.DataSet.CaseSensitive)) {
    this.CaseSensitive = table.CaseSensitive;
}
if ((table.Locale.ToString() != table.DataSet.Locale.ToString())) {
    this.Locale = table.Locale;
}
if ((table.Namespace != table.DataSet.Namespace)) {
    this.Namespace = table.Namespace;
}
this.Prefix = table.Prefix;
this.MinimumCapacity = table.MinimumCapacity;
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
protected КлиентиDataTable(global::System.Runtime.Serialization.SerializationInfo
info, global::System.Runtime.Serialization.StreamingContext context) :
    base(info, context) {
    this.InitVars();
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
public global::System.Data.DataColumn КодКлієнтаColumn {
    get {
        return this.columnКодКлієнта;
    }
}

```



```
}  
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]  
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.  
TypedDataSetGenerator", "4.0.0.0")]  
public global::System.Data.DataColumn НайменуванняColumn {  
    get {  
        return this.columnНайменування;  
    }  
}  
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]  
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.  
TypedDataSetGenerator", "4.0.0.0")]  
public global::System.Data.DataColumn КонтактиColumn {  
    get {  
        return this.columnКонтакти;  
    }  
}  
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]  
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.  
TypedDataSetGenerator", "4.0.0.0")]  
public global::System.Data.DataColumn ПриміткаColumn {  
    get {  
        return this.columnПримітка;  
    }  
}  
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]  
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.  
TypedDataSetGenerator", "4.0.0.0")]
```

```

[global::System.ComponentModel.Browsable(false)]
public int Count {
    get {
        return this.Rows.Count;
    }
}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
public КлієнтиRow this[int index] {
    get {
        return ((КлієнтиRow)(this.Rows[index]));
    }
}

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
public event КлієнтиRowChangeEventHandler КлієнтиRowChanging;
[global::System.CodeDom.Compiler.GeneratedCodeAttribute
"System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]
public event КлієнтиRowChangeEventHandler КлієнтиRowChanged;
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
public event КлієнтиRowChangeEventHandler КлієнтиRowDeleting;
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
public event КлієнтиRowChangeEventHandler КлієнтиRowDeleted;
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.

```



```

TypedDataSetGenerator", "4.0.0.0")]
public void AddКлієнтиRow(КлієнтиRow row) {
    this.Rows.Add(row);
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
public КлієнтиRow AddКлієнтиRow(string Найменування, string Контакти, string
Примітка) {
    КлієнтиRow rowКлієнтиRow = ((КлієнтиRow)(this.NewRow()));
    object[] columnValuesArray = new object[] {
        null,
        Найменування,
        Контакти,
        Примітка};
    rowКлієнтиRow.ItemArray = columnValuesArray;
    this.Rows.Add(rowКлієнтиRow);
    return rowКлієнтиRow;
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
public КлієнтиRow FindByКодКлієнта(int КодКлієнта) {
    return ((КлієнтиRow)(this.Rows.Find(new object[] {
        КодКлієнта})));
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.

```

```

TypedDataSetGenerator", "4.0.0.0")]
public override global::System.Data.DataTable Clone() {
КлієнтиDataTable cIn = ((КлієнтиDataTable)(base.Clone()));
cIn.InitVars();
return cIn;
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
protected override global::System.Data.DataTable CreateInstance() {
return new КлієнтиDataTable();
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
internal void InitVars() {
this.columnКодКлієнта = base.Columns["КодКлієнта"];
this.columnНайменування = base.Columns["Найменування"];
this.columnКонтакти = base.Columns["Контакти"];
this.columnПримітка = base.Columns["Примітка"];
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
private void InitClass() {
this.columnКодКлієнта = new global::System.Data.DataColumn "КодКлієнта",
typeof(int), null, global::System.Data.MappingType.Element);
base.Columns.Add(this.columnКодКлієнта);
}

```



```

this.columnНайменування = new global::System.Data.DataColumn
"Найменування", typeof(string), null, global::System.Data.MappingType.Element);
base.Columns.Add(this.columnНайменування);
this.columnКонтакти = new global::System.Data.DataColumn "Контакти",
typeof(string), null, global::System.Data.MappingType.Element);
base.Columns.Add(this.columnКонтакти);
this.columnПримітка = new global::System.Data.DataColumn "Примітка",
typeof(string), null, global::System.Data.MappingType.Element);
base.Columns.Add(this.columnПримітка);
this.Constraints.Add(new global::System.Data.UniqueConstraint "Constraint1", new
global::System.Data.DataColumn[] {
    this.columnКодКлієнта}, true));
this.columnКодКлієнта.AutoIncrement = true;
this.columnКодКлієнта.AutoIncrementSeed = -1;
this.columnКодКлієнта.AutoIncrementStep = -1;
this.columnКодКлієнта.AllowDBNull = false;
this.columnКодКлієнта.ReadOnly = true;
this.columnКодКлієнта.Unique = true;
this.columnНайменування.AllowDBNull = false;
this.columnНайменування.MaxLength = 100;
this.columnКонтакти.MaxLength = 200;
this.columnПримітка.MaxLength = 200;
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
public КлієнтиRow NewКлієнтиRow() {
return ((КлієнтиRow)(this.NewRow()));
}

```

```

}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
protected override global::System.Data.DataRow
NewRowFromBuilder(global::System.Data.DataRowBuilder builder) {
return new КлієнтиRow(builder);
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
protected override global::System.Type GetRowType() {
return typeof(КлієнтиRow);
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
protected override void
OnRowChanged(global::System.Data.DataRowChangeEventArgs e) {
base.OnRowChanged(e);
if ((this.КлієнтиRowChanged != null)) {
this.КлієнтиRowChanged(this, new
КлієнтиRowChangeEvent(((КлієнтиRow)(e.Row)), e.Action));
}
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]

```



```

protected override void
OnRowChanging(global::System.Data.DataRowChangeEventArgs e) {
    base.OnRowChanging(e);
    if ((this. КлієнтиRowChanging != null)) {
        this. КлієнтиRowChanging(this, new
        КлієнтиRowChangeEvent(((КлієнтиRow)(e.Row)), e.Action));
    }
}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
protected override void
OnRowDeleted(global::System.Data.DataRowChangeEventArgs e) {
    base.OnRowDeleted(e);
    if ((this. КлієнтиRowDeleted != null)) {
        this. КлієнтиRowDeleted(this, new
        КлієнтиRowChangeEvent(((КлієнтиRow)(e.Row)), e.Action));
    }
}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
protected override void
OnRowDeleting(global::System.Data.DataRowChangeEventArgs e) {
    base.OnRowDeleting(e);
    if ((this. КлієнтиRowDeleting != null)) {
        this. КлієнтиRowDeleting(this, new
        КлієнтиRowChangeEvent(((КлієнтиRow)(e.Row)), e.Action));
    }
}

```

```
}  
}
```

```
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
```

```
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.  
TypedDataSetGenerator", "4.0.0.0")]
```

```
public static global::System.Xml.Schema.XmlSchemaComplexType
```

```
GetTypedTableSchema(global::System.Xml.Schema.XmlSchemaSet xs) {
```

```
global::System.Xml.Schema.XmlSchemaComplexType type = new
```

```
global::System.Xml.Schema.XmlSchemaComplexType();
```

```
global::System.Xml.Schema.XmlSchemaSequence sequence = new
```

```
global::System.Xml.Schema.XmlSchemaSequence();
```

```
MyDataSet ds = new MyDataSet();
```

```
global::System.Xml.Schema.XmlSchemaAny any1 = new
```

```
global::System.Xml.Schema.XmlSchemaAny();
```

```
any1.Namespace = "http://www.w3.org/2001/XMLSchema";
```

```
any1.MinOccurs = new decimal(0);
```

```
any1.MaxOccurs = decimal.MaxValue;
```

```
any1.ProcessContents =
```

```
global::System.Xml.Schema.XmlSchemaContentProcessing.Iax;
```

```
sequence.Items.Add(any1);
```

```
global::System.Xml.Schema.XmlSchemaAny any2 = new
```

```
global::System.Xml.Schema.XmlSchemaAny();
```

```
any2.Namespace = "urn:schemas-microsoft-com:xml-diffgram-v1";
```

```
any2.MinOccurs = new decimal(1);
```

```
any2.ProcessContents =
```

```
global::System.Xml.Schema.XmlSchemaContentProcessing.Iax;
```

```
sequence.Items.Add(any2);
```

```
global::System.Xml.Schema.XmlSchemaAttribute attribute1 = new
```



```

global::System.Xml.Schema.XmlSchemaAttribute();
attribute1.Name = "namespace";
attribute1.FixedValue = ds.Namespace;
type.Attributes.Add(attribute1);
global::System.Xml.Schema.XmlSchemaAttribute attribute2 = new
global::System.Xml.Schema.XmlSchemaAttribute();
attribute2.Name = "tableName";
attribute2.FixedValue = "КлієнтиDataTable";
type.Attributes.Add(attribute2);
type.Particle = sequence;
global::System.Xml.Schema.XmlSchema dsSchema = ds.GetSchemaSerializable();
if (xs.Contains(dsSchema.TargetNamespace)) {
    global::System.IO.MemoryStream s1 = new global::System.IO.MemoryStream();
    global::System.IO.MemoryStream s2 = new global::System.IO.MemoryStream();
    try {
        global::System.Xml.Schema.XmlSchema schema = null;
        dsSchema.Write(s1);
        for (global::System.Collections.IEnumerator schemas =
xs.Schemas(dsSchema.TargetNamespace).GetEnumerator(); schemas.MoveNext(); )
        {
            schema = ((global::System.Xml.Schema.XmlSchema)(schemas.Current));
            s2.SetLength(0);
            schema.Write(s2);
            if ((s1.Length == s2.Length)) {
                s1.Position = 0;
                s2.Position = 0;
                for (; ((s1.Position != s1.Length)
&& (s1.ReadByte() == s2.ReadByte())); ) {

```

```
;
}
```

```
if ((s1.Position == s1.Length)) {
    return type;
}
```

///Represents the strongly named DataTable class.

///</summary>

[global::System.Serializable()]

[global::System.Xml.Serialization.XmlSchemaProviderAttribute("GetTypedTableSchema")]

public partial class МісяЗберіганняDataTable :

global::System.Data.TypedTableBase<МісяЗберіганняRow> {

private global::System.Data.DataColumn columnКодМісяЗберігання;

private global::System.Data.DataColumn columnНайменування;

private global::System.Data.DataColumn columnПримітка;

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]

public МісяЗберіганняDataTable() {

this.TableName = "МісяЗберігання";

this.BeginInit();

this.InitClass();

this.EndInit();

}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.TypedDataSetGenerator", "4.0.0.0")]

internal МісяЗберіганняDataTable(global::System.Data.DataTable table) {


```

this.TableName = table.TableName;
if ((table.CaseSensitive != table.DataSet.CaseSensitive)) {
    this.CaseSensitive = table.CaseSensitive;
}
if ((table.Locale.ToString() != table.DataSet.Locale.ToString())) {
    this.Locale = table.Locale;
}
if ((table.Namespace != table.DataSet.Namespace)) {
    this.Namespace = table.Namespace;
}
this.Prefix = table.Prefix;
this.MinimumCapacity = table.MinimumCapacity;
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
protected
МісцяЗберіганняDataTable(global::System.Runtime.Serialization.SerializationInfo
info, global::System.Runtime.Serialization.StreamingContext context) :
    base(info, context) {
    this.InitVars();
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
public global::System.Data.DataColumn КодМісцяЗберіганняColumn {
    get {
        return this.columnКодМісцяЗберігання;
    }
}

```

```

}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
public global::System.Data.DataColumn НайменуванняColumn {
get {
return this.columnНайменування;
}
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
public global::System.Data.DataColumn ПриміткаColumn {
get {
return this.columnПримітка;
}
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
[global::System.ComponentModel.Browsable(false)]
public int Count {
get {
return this.Rows.Count;
}
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.

```



```

TypedDataSetGenerator", "4.0.0.0")]
public МісяЗберіганняRow this[int index] {
    get {
        return ((МісяЗберіганняRow)(this.Rows[index]));
    }
}

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
public event МісяЗберіганняRowChangeEventHandler
МісяЗберіганняRowChanging;
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
public event МісяЗберіганняRowChangeEventHandler
МісяЗберіганняRowChanged;
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
public event МісяЗберіганняRowChangeEventHandler
МісяЗберіганняRowDeleting;
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
public event МісяЗберіганняRowChangeEventHandler
МісяЗберіганняRowDeleted;
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
public void AddМісяЗберіганняRow(МісяЗберіганняRow row) {
    this.Rows.Add(row);
}

```

```

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
public МісцяЗберіганняRow AddМісцяЗберіганняRow(string Найменування,
string Примітка) {
    МісцяЗберіганняRow rowМісцяЗберіганняRow =
    ((МісцяЗберіганняRow)(this.NewRow()));
    object[] columnValuesArray = new object[] {
        null,
        Найменування,
        Примітка};
    rowМісцяЗберіганняRow.ItemArray = columnValuesArray;
    this.Rows.Add(rowМісцяЗберіганняRow);
    return rowМісцяЗберіганняRow;
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
public МісцяЗберіганняRow FindByКодМісцяЗберігання (int
КодМісцяЗберігання) {
    return ((МісцяЗберіганняRow)(this.Rows.Find(new object[] {
        КодМісцяЗберігання})));
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
public override global::System.Data.DataTable Clone() {
    МісцяЗберіганняDataTable cIn = ((МісцяЗберіганняDataTable)(base.Clone()));

```



```

cIn.InitVars();
return cIn;
}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
protected override global::System.Data.DataTable CreateInstance() {
return new МісцяЗберіганняDataTable();
}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
internal void InitVars() {
this.columnКодМісцяЗберігання = base.Columns["КодМісцяЗберігання"];
this.columnНайменування = base.Columns["Найменування"];
this.columnПримітка = base.Columns["Примітка"];
}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
private void InitClass() {
this.columnКодМісцяЗберігання = new global::System.Data.DataColumn
"КодМісцяЗберігання", typeof(int), null,
global::System.Data.MappingType.Element);
base.Columns.Add(this.columnКодМісцяЗберігання);
this.columnНайменування = new global::System.Data.DataColumn
("Найменування", typeof(string), null, global::System.Data.MappingType.Element);
base.Columns.Add(this.columnНайменування);
}

```

```

this.columnПримітка = new global::System.Data.DataColumn("Примітка",
typeof(string), null, global::System.Data.MappingType.Element);
base.Columns.Add(this.columnПримітка);
this.Constraints.Add(new global::System.Data.UniqueConstraint("Constraint1", new
global::System.Data.DataColumn[] {
    this.columnКодМісяЗберігання}, true));
this.columnКодМісяЗберігання.AutoIncrement = true;
this.columnКодМісяЗберігання.AutoIncrementSeed = -1;
this.columnКодМісяЗберігання.AutoIncrementStep = -1;
this.columnКодМісяЗберігання.AllowDBNull = false;
this.columnКодМісяЗберігання.ReadOnly = true;
this.columnКодМісяЗберігання.Unique = true;
this.columnНайменування.AllowDBNull = false;
this.columnНайменування.MaxLength = 100;
this.columnПримітка.MaxLength = 100;
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
public МісяЗберіганняRow NewМісяЗберіганняRow() {
return ((МісяЗберіганняRow)(this.NewRow()));
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
protected override global::System.Data.DataRow
NewRowFromBuilder(global::System.Data.DataRowBuilder builder) {
return new МісяЗберіганняRow(builder);
}

```



```

}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
protected override global::System.Type GetRowType() {
return typeof(МісцяЗберіганняRow);
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
protected override void
OnRowChanged(global::System.Data.DataRowChangeEventArgs e) {
if ((this. МісцяЗберіганняRowChanged != null)) {
this. МісцяЗберіганняRowChanged(this, new
МісцяЗберіганняRowChangeEvent(((МісцяЗберіганняRow)(e.Row)), e.Action));
}
}
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
protected override void
OnRowChanging(global::System.Data.DataRowChangeEventArgs e) {
base.OnRowChanging(e);
if ((this. МісцяЗберіганняRowChanging != null)) {
this. МісцяЗберіганняRowChanging(this, new
МісцяЗберіганняRowChangeEvent(((МісцяЗберіганняRow)(e.Row)), e.Action));
}
}
}

```

```

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
protected override void
OnRowDeleted(global::System.Data.DataRowChangeEventArgs e) {
    base.OnRowDeleted(e);
    if ((this. МісяцьЗберіганняRowDeleted != null)) {
        this. МісяцьЗберіганняRowDeleted(this, new
        МісяцьЗберіганняRowChangeEvent(((МісяцьЗберіганняRow)(e.Row)), e.Action));
    }
}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
protected override void
OnRowDeleting(global::System.Data.DataRowChangeEventArgs e) {
    base.OnRowDeleting(e);
    if ((this. МісяцьЗберіганняRowDeleting != null)) {
        this. МісяцьЗберіганняRowDeleting(this, new
        МісяцьЗберіганняRowChangeEvent(((МісяцьЗберіганняRow)(e.Row)), e.Action));
    }
}

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
public void RemoveМісяцьЗберіганняRow(МісяцьЗберіганняRow row) {
    this.Rows.Remove(row);
}

```



```

[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Data.Design.
TypedDataSetGenerator", "4.0.0.0")]
public static global::System.Xml.Schema.XmlSchemaComplexType
GetTypedTableSchema(global::System.Xml.Schema.XmlSchemaSet xs) {
global::System.Xml.Schema.XmlSchemaComplexType type = new
global::System.Xml.Schema.XmlSchemaComplexType();
global::System.Xml.Schema.XmlSchemaSequence sequence = new
global::System.Xml.Schema.XmlSchemaSequence();
MyDataSet ds = new MyDataSet();
global::System.Xml.Schema.XmlSchemaAny any1 = new
global::System.Xml.Schema.XmlSchemaAny();
any1.Namespace = "http://www.w3.org/2001/XMLSchema";
any1.MinOccurs = new decimal(0);
any1.MaxOccurs = decimal.MaxValue;
any1.ProcessContents =
global::System.Xml.Schema.XmlSchemaContentProcessing.Lax;
sequence.Items.Add(any1);
global::System.Xml.Schema.XmlSchemaAny any2 = new
global::System.Xml.Schema.XmlSchemaAny();
any2.Namespace = "urn:schemas-microsoft-com:xml-diffgram-v1";
any2.MinOccurs = new decimal(1);
any2.ProcessContents =
global::System.Xml.Schema.XmlSchemaContentProcessing.Lax;
sequence.Items.Add(any2);
global::System.Xml.Schema.XmlSchemaAttribute attribute1 = new
global::System.Xml.Schema.XmlSchemaAttribute();
attribute1.Name = "namespace";

```

```
attribute1.FixedValue = ds.Namespace;
type.Attributes.Add(attribute1);
global::System.Xml.Schema.XmlSchemaAttribute attribute2 = new
global::System.Xml.Schema.XmlSchemaAttribute();
attribute2.Name = "tableTypeName";
attribute2.FixedValue = "МісцяЗберіганняDataTable";
type.Attributes.Add(attribute2);
type.Particle = sequence;
global::System.Xml.Schema.XmlSchema dsSchema = ds.GetSchemaSerializable();
if (xs.Contains(dsSchema.TargetNamespace)) {
    global::System.IO.MemoryStream s1 = new global::System.IO.MemoryStream();
    global::System.IO.MemoryStream s2 = new global::System.IO.MemoryStream();
    try {
        global::System.Xml.Schema.XmlSchema schema = null;
        dsSchema.Write(s1);
        for (global::System.Collections.IEnumerator schemas =
xs.Schemas(dsSchema.TargetNamespace).GetEnumerator(); schemas.MoveNext(); )
        {
            schema = ((global::System.Xml.Schema.XmlSchema)(schemas.Current));
            s2.SetLength(0);
            schema.Write(s2);
            if ((s1.Length == s2.Length)) {
                s1.Position = 0;
                s2.Position = 0;
                for (; ((s1.Position != s1.Length)
&& (s1.ReadByte() == s2.ReadByte())); ) {
                    ;
                }
            }
        }
    }
}
```



```
if ((s1.Position == s1.Length)) {  
    return type;  
}  
}  
}  
finally {  
    if ((s1 != null)) {  
        s1.Close();  
    }  
    if ((s2 != null)) {  
        s2.Close()  
    }  
}
```

