МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТЕХНОЛОГІЙ

ННІ/факультет	Інформаційних технологій
Кафедра	Інформатики і прикладного програмного забезпечення
Спеціальність	Інженерія програмного забезпечення
Форма навчання	Денна

КВАЛІФІКАЦІЙНА БАКАЛАВРСЬКА РОБОТА

Дегтярьова Артема Олександровича
(прізвище, ім'я, по батькові здобувача)

на тему
Розробка програмного забезпечення організації туристичних турів
(повна назва теми)

за матеріалами праць провідних спеціалістів з розробки ПЗ та проектування БД
(повна назва бази дослідження)

науковий керівник к.т.н., _____ Медведєв Д.Г.
(наук. ступінь, вчене звання) *(підпис)* *(прізвище, ініціали)*

Робота допущена до захисту в ЕК
Протокол засідання кафедри
від 11.06.2025р. № 12
Завідувач кафедри _____
(підпис)

д.т.н., професор Зеленський О.С.
Наук. ступень, вчене звання *Ініціали, прізвище*

Кривий Ріг – 2025

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТЕХНОЛОГІЙ

ННІ/факультет	Інформаційних технологій
Кафедра	Інформатики і прикладного програмного забезпечення
Спеціальність	Інженерія програмного забезпечення
Форма навчання	Денна

«ЗАТВЕРДЖУЮ»
Завідувач кафедри _____ Зеленський О.С.
(підпис) (Прізвище, ініціали)
« 11» червня 2025 року

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ МАГІСТЕРСЬКУ РОБОТУ

1. Тема роботи «Розробка програмного забезпечення організації туристичних турів»

Керівник роботи к.т.н., Медведєв Д.Г.
затверджені наказом закладу вищої освіти від «4» квітня 2025 р. № 222-ст

2. Строк подання здобувачем роботи до «9» червня 2025 р.

3. Зміст кваліфікаційної роботи, об’єкт, предмет та мета дослідження:

Розділ 1. Постановка задачі

Розділ 2. Розробка алгоритму розв’язання задачі

Розділ 3. Організація інформаційного забезпечення

Розділ 4. Розробка програмного забезпечення

Об’єкт дослідження: процес обробки, візуалізації та управління інформацією про туристичні тури

Предмет дослідження: програмне забезпечення для автоматизації діяльності туристичних фірм

Мета кваліфікаційної роботи: розробити десктопне програмне забезпечення для ефективної організації, представлення та аналізу даних туристичних турів

5. Дата видачі завдання «4» квітня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів МДР	Строк виконання етапів роботи	Відмітка керівника про виконання етапів (дата, підпис)
1	Підготовка розділу 1	04.04.2025-13.04.2025	
2	Підготовка розділу 2	14.04.2025-26.04.2025	
3.	Підготовка розділу 3	27.04.2025-15.05.2025	
4	Підготовка розділу 4	16.05.2025-08.06.2025	
5	Реєстрація завершеної кваліфікаційної роботи	09.06.2025	Реєстраційний №___ «09»червня 2025 р.
6	Отримання відгуку від наукового керівника	10.06.2025	
7	Подання кваліфікаційної роботи на перегляд завідувачу кафедри	11.06.2025	
8	Отримання зовнішньої рецензії	12.06.2025	
9	Попередній захист кваліфікаційної роботи на кафедрі	13.06.2025	
10	Підготовка до захисту в ЕК	16.06.2025-21.06.2025	

Завдання підготував науковий керівник _____ __Медведєв Д.Г.__
(підпис) (прізвище та ініціали)

Завдання одержав _____ __Дегтярьов А.О.__
(підпис) (прізвище та ініціали)

А Н О Т А Ц І Я

на кваліфікаційну бакалаврську роботу

«Розробка програмного забезпечення організації туристичних турів»

Дегтярьова Артема Олександровича

Кваліфікаційна бакалаврська робота на здобуття освітньо-кваліфікаційного рівня бакалавра зі спеціальності 121 «Інженерія програмного забезпечення» – Державний університет економіки і технологій – Кривий Ріг, 2025.

У бакалаврській дипломній роботі розроблено програмне забезпечення для автоматизації процесу організації туристичних турів.

Додаток забезпечує введення, редагування та візуальне представлення даних про тури, готелі та країни.

Програма надає можливість налаштування вигляду таблиць, зокрема шрифтів і кольорів заголовків та тексту. Реалізовано експорт даних у форматі HTML із підтримкою стилів, а також побудову графіків і гістограм за допомогою OpenGL.

Програмний продукт розроблено мовою програмування C# з використанням бібліотек Windows Forms, OpenGL (через Tao Framework) для графічної візуалізації та технологій HTML/JavaScript для формування звітів.

Інтерфейс додатку включає головне меню, діалогові вікна для налаштувань і форму перегляду інформації про програму.

Ключові слова: ТУРИЗМ, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, C#, OPENGL, HTML, ІНТЕРФЕЙС, ВІЗУАЛІЗАЦІЯ, ТАБЛИЦЯ, ЗВІТ, ЕКСПОРТ.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

C#	Мова програмування високого рівня, розроблена компанією Microsoft як частина платформи .NET. C# підтримує об'єктно-орієнтоване програмування, має потужну типізацію, багату стандартну бібліотеку та широкі можливості для створення графічних інтерфейсів користувача, роботи з базами даних, мережею та візуалізацією.
Windows Forms	Технологія побудови графічного інтерфейсу користувача (GUI) у Windows-додатках на платформі .NET. Дозволяє створювати форми, кнопки, меню та інші елементи інтерфейсу. Підтримує подієву модель програмування та забезпечує швидкий доступ до засобів Windows.
OpenGL	(англ. Open Graphics Library) — кросплатформовий програмний інтерфейс (API) для створення дво- та тривимірної графіки. Широко використовується в ігровій індустрії, візуалізації даних, CAD-системах тощо. Забезпечує низькорівневий доступ до відеокарти для побудови графіків, гістограм, діаграм.
HTML	(англ. HyperText Markup Language) — мова розмітки документів, що використовується для створення веб-сторінок. У програмі застосовується для формування структурованих, стилізованих звітів про туристичні тури.
ADO.NET	Компонент .NET Framework, що забезпечує доступ до джерел даних, зокрема до баз даних. Забезпечує об’єктно-орієнтований підхід до роботи з таблицями, запитами та зв'язками.
XML	(англ. Extensible Markup Language) — розширювана мова розмітки, яка використовується для зберігання та передачі структурованих даних. Може застосовуватись для збереження налаштувань програми або даних користувача.

ЗМІСТ

ВСТУП 7
РОЗДІЛ 1 ПОСТАНОВКА ЗАДАЧІ 9
1.1 Аналіз вимог та характеристика задач 9
1.2. Огляд існуючих додатків 13
1.3. Вхідна та вихідна інформація 17
РОЗДІЛ 2 РОЗРОБКА АЛГОРИТМУ РОЗВ'ЯЗАННЯ ЗАДАЧІ 19
2.1. Загальна характеристика задач 19
2.2. Алгоритм виконання роботи 21
РОЗДІЛ 3 ОРГАНІЗАЦІЯ ІНФОРМАЦІЙНОГО ЗАБЕЗПЕЧЕННЯ 27
3.1. Загальна характеристика інформаційного забезпечення 27
3.2. Побудова системи класифікації та кодування 30
3.3. Характеристика та вимоги до бази даних. 34
3.4. Структура бази даних. 38
РОЗДІЛ 4 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ЗАДАЧІ 55
4.1. Проектування розробки програмного забезпечення 55
4.2. Опис програмного забезпечення «Tyrfirma» 57
ВИСНОВКИ 76
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 78
ДОДАТКИ 79

ВСТУП

Сучасний етап розвитку інформаційних технологій супроводжується стрімкою діджиталізацією майже всіх сфер людської діяльності. Особливо це стосується сфери обслуговування та туризму — однієї з найбільш динамічних галузей світової економіки. В умовах глобальної конкуренції, нестабільності ринку та високих вимог з боку клієнтів, туристичні компанії змушені постійно вдосконалювати свої підходи до організації діяльності. Одним із ключових чинників успіху є впровадження сучасного програмного забезпечення, яке дозволяє автоматизувати рутинні процеси, оптимізувати управління даними та покращити якість обслуговування клієнтів.

Автоматизація діяльності туристичних фірм є важливим кроком до підвищення ефективності їх функціонування. Особливо актуальним стає створення програмного продукту, що дозволить інтегрувати в єдину систему облік туристичних послуг, візуалізацію даних (зокрема у вигляді графіків і діаграм), а також забезпечить зручний та інтуїтивно зрозумілий інтерфейс.

На сьогодні більшість програм для туристичних агентств мають вузьку спеціалізацію, складні у використанні або не відповідають сучасним вимогам до дизайну, гнучкості та інтерактивності. Тому актуальним завданням є розробка нового програмного забезпечення, що поєднує в собі функціональність, зручність, підтримку мультимедіа, багатомовність і можливість інтеграції з іншими системами.

Метою даної дипломної роботи є розробка інтерактивного програмного забезпечення для автоматизації основних процесів туристичної фірми, яке б дозволяло обробляти інформацію про тури, країни, готелі, візуалізувати статистичні дані, формувати HTML-звіти, а також представляти інформацію у вигляді 3D-графіки.

Для досягнення цієї мети поставлено такі основні завдання:

- провести аналіз сучасних інформаційних систем для туристичної галузі;

- визначити функціональні та нефункціональні вимоги до розроблюваного програмного забезпечення;
- спроектувати архітектуру ПЗ з урахуванням можливості масштабування та подальшої підтримки;
- реалізувати програму з використанням мови програмування C++ та бібліотеки OpenGL для 3D-візуалізації;
- забезпечити генерацію HTML-звітів для представлення результатів у браузері;
- реалізувати багатомовний інтерфейс і модуль довідки з використанням CHM-файлів;
- протестувати готовий продукт та оцінити його функціональність.

Об’єктом дослідження є процес автоматизації управління даними в туристичних компаніях. Предметом дослідження виступають методи, засоби та алгоритми, які використовуються для створення прикладного програмного забезпечення в туристичній сфері.

Практичне значення роботи полягає у створенні реального інструменту, який може бути використаний у діяльності малих та середніх туристичних компаній для полегшення процесу взаємодії з клієнтами, ведення обліку та прийняття управлінських рішень.

У процесі виконання дипломної роботи були застосовані такі інструменти: мова програмування C# графічна бібліотека OpenGL, середовище розробки Visual Studio, засоби створення HTML-документів та довідкової системи у форматі CHM. Програма розроблена у вигляді класичного Windows-додатка з підтримкою інтуїтивно зрозумілого графічного інтерфейсу та інтерактивних елементів.

Таким чином, дана дипломна робота є актуальною як з точки зору сучасних технологічних вимог, так і з точки зору потреб реального ринку туристичних послуг. Вона демонструє можливості застосування сучасних мов програмування та бібліотек для вирішення прикладних задач у сфері автоматизації та діджиталізації бізнесу.

РОЗДІЛ 1
ПОСТАНОВКА ЗАДАЧІ

1.1 Аналіз вимог та характеристика задач

Розберемо загальну постановку задачі.

Автоматизація діяльності туристичних фірм потребує спеціалізованих програмних засобів, які забезпечують зручне управління інформацією про туристичні послуги, формування звітів, візуалізацію аналітичних даних та гнучкість у налаштуванні зовнішнього вигляду інтерфейсу.

З огляду на це, завданням розробки є створення десктопного програмного забезпечення з графічним інтерфейсом, яке реалізує вивід і редагування даних про готелі, країни, тури, формування HTML-звітів (див. Додаток А), а також графічну візуалізацію статистики за допомогою OpenGL. [4]

Програмне забезпечення повинно мати можливість:

- зберігати й обробляти дані про туристичні тури, готелі, країни;
- налаштовувати зовнішній вигляд таблиць (шрифти, кольори заголовків і тексту);
- створювати HTML-звіти з підтримкою кольорового форматування;
- візуалізувати статистику у вигляді графіків та гістограм (OpenGL) (див. Додаток Б);
- забезпечити зручну взаємодію через меню, діалогові вікна та головне вікно програми;
- відображати інформацію про авторські права (форма «Про програму») (див. Додаток В);
- бути стабільним, інтуїтивно зрозумілим і простим у користуванні.

Створимо класифікацію та формалізацію вимог. Було прийнято рішення розділити вимоги на два типи:

1. Функціональні вимоги:
    - Інтерфейс введення/редагування даних.

- Форматування таблиць.
- Графічна візуалізація.
- Експорт даних у HTML.
- Довідкова система.
- Меню керування.

2. Нефункціональні вимоги:
   - Зручність у користуванні (usability).
   - Можливість розширення.
   - Портативність.
   - Ефективність.
   - Надійність.

Перейдемо до вибору технологій:

- Мова програмування: C# — для реалізації основної логіки та GUI (Windows Forms). [1]
- OpenGL — для побудови графіків і гістограм.
- HTML — для експорту таблиць і звітів у форматі, зручному для перегляду.
- Системні діалогові вікна Windows — для вибору шрифтів, кольорів тощо.

Користувач взаємодіє з головним вікном, де доступні всі функції (Рис. 1.1.).

Через меню можна відкривати діалогові вікна для налаштування вигляду, запускати побудову графіків, переглядати HTML-звіти тощо.

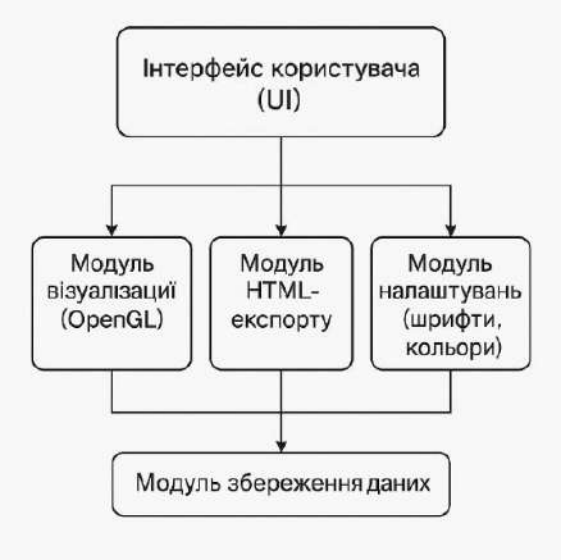


Рис. 1.1. Архітектура взаємодії модулів програми (схема)

Інтерфейс програми реалізований у вигляді головного вікна, в якому зосереджені основні елементи керування.

Зокрема, у верхній частині вікна розміщено меню, що містить пункти для виклику функціоналу: додавання та редагування даних, побудова графіків, відкриття діалогів налаштування шрифтів і кольорів, формування HTML-звітів тощо.

Центральна частина головного вікна містить таблицю, в якій відображається інформація про готелі, тури, країни та інші сутності.

Таблиця підтримує налаштування кольорів заголовків, фону та шрифтів, що дозволяє користувачеві адаптувати вигляд інтерфейсу під власні вподобання.

У цілому, головне вікно забезпечує інтуїтивно зрозумілу взаємодію з програмою та дозволяє виконувати всі основні дії без необхідності переходу до окремих підрозділів чи вкладок.

Структура головного вікна зображена на рисунку нижче (Рис. 1.2).

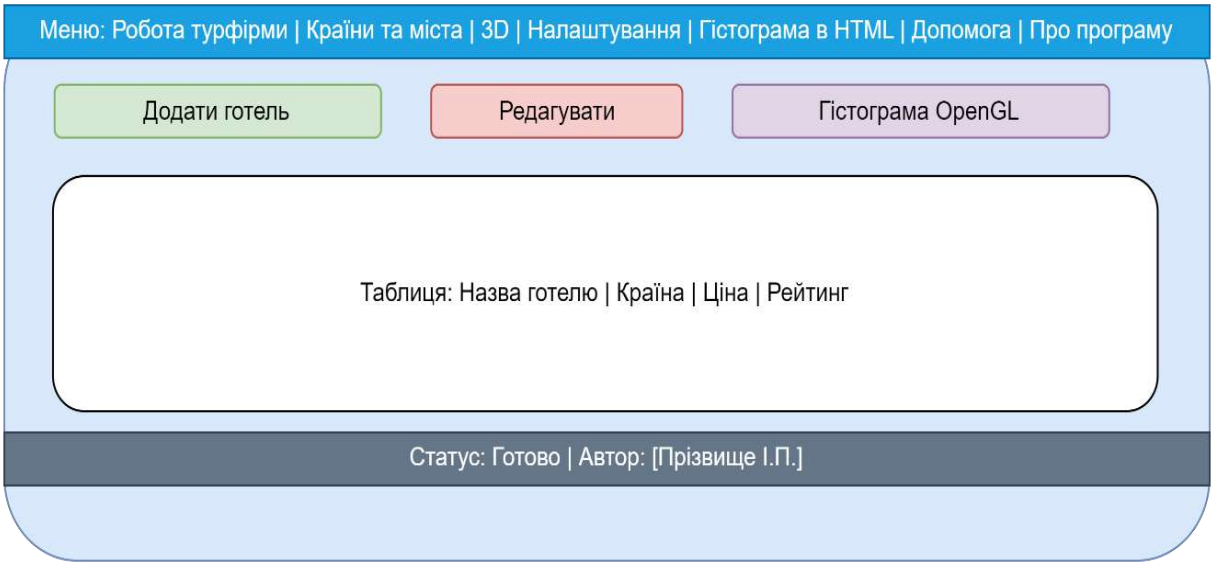


Рис. 1.2. Приклад головного вікна з меню керування та кнопками функцій

Для кращого розуміння функціональних можливостей програмного забезпечення та структури взаємодії користувача з основними модулями системи було побудовано діаграму варіантів використання (use-case діаграму).

На ній зображено основні дії, які може виконувати користувач у межах програми.

Зокрема, користувач має змогу:

- переглядати список готелів та турів;
- додавати нові записи до бази даних;
- редагувати та видаляти наявну інформацію;
- будувати графіки з використанням OpenGL; [4]
- змінювати вигляд таблиць (шрифт, колір);
- формувати звіти у форматі HTML;
- переглядати довідкову інформацію та відомості про програму.

Дана діаграма ілюструє роль користувача як єдиного активного учасника системи, який взаємодіє з кількома варіантами використання через інтерфейс програми.

Це дозволяє систематизувати вимоги до функціоналу ПЗ та забезпечує чітке розуміння архітектури з точки зору користувача.

Структура взаємодії представлена на рисунку нижче (Рис. 1.3).

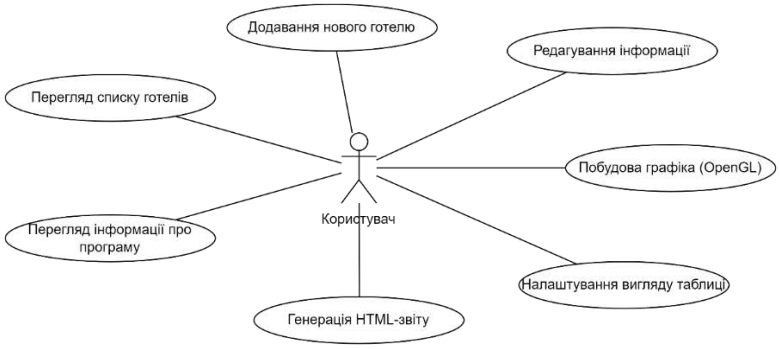


Рис. 1.3. Діаграма варіантів використання (use-case діаграма)

Таким чином, розроблене програмне забезпечення орієнтоване на зручність, наочність і ефективну взаємодію з користувачем, що є ключовим у туристичній галузі.

1.2. Огляд існуючих додатків

Існує безліч застосунків турфірм. Розглянемо декілька web-сайтів, для визначення переваг та недоліків.

Перший web-сайт, який ми розглянемо – це сайт турфірми «Поїхали з нами».

Посилання: https://www.poehalisnami.ua/ua

Слід зазначити що, першою перевагою є те, що вигляд головної сторінки адаптується під тему браузеру (Рис. 1.4.).

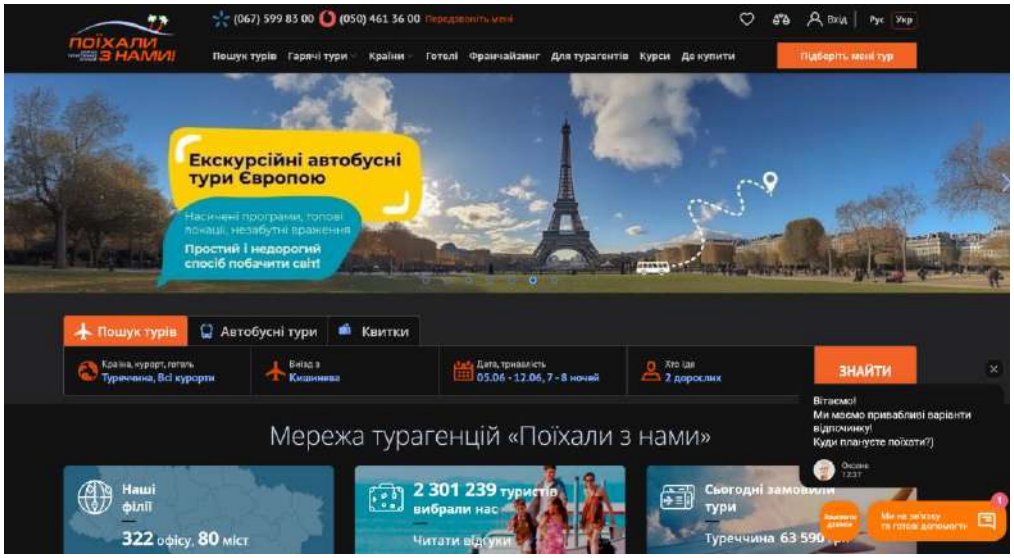


Рис. 1.4. Вигляд головної сторінки сайту турфірми «Поїхали з нами»

Ось які переваги можна визначити:

- Швидкий пошук турів:

  Усі головні параметри (країна, місто вильоту, дати, тривалість, кількість осіб) — у зручному верхньому блоці.

  Можна легко змінити фільтри без перезавантаження сторінки.
- Зручна аналітика цін:

  Таблиця з мінімальними цінами по днях — наочно показує, коли вигідніше їхати.

  Найнижча ціна виділена кольором (25 120 грн) — це добре видно.
- Велика база варіантів:

  716 готелів у результатах пошуку, з яких 393 — рекомендовані (довіра до вибору).
- Підтримка клієнтів:

  Онлайн-чат, кнопка «Ми на зв’язку», запит на зворотній дзвінок — це швидкий зв’язок із менеджером.
- Наявність франшизи та курсів:

  Сайт має розділ «Франчайзинг» і «Курси» — не лише для туристів, а й для тих, хто хоче працювати у сфері туризму.

Серед достатньої кількості переваг, даний сайт має також і недоліки:

- Основна мова сайту — російська:

  Незважаючи на можливість обрати українську, переважна частина сайту все одно залишається російською.
- Візуально перевантажений інтерфейс:

  Надто багато інформації та дрібного тексту — не кожному зручно швидко орієнтуватися.
- Надокучливі спливаючі вікна:

  Автоматичне повідомлення від менеджера (чат) може заважати користувачу.
- Відсутність розширених фільтрів:

Немає можливості одразу обрати тип харчування, рейтинг готелю, зірковість, відгуки тощо (потрібно клікати далі).

- Дизайн дещо застарілий:

  На фоні сучасних тревел-сервісів сайт виглядає не надто сучасно — бракує анімації, адаптивності, інтерактивності.

Як підсумок слід зазначити, що сайт зручний для швидкого підбору турів, особливо якщо вам потрібна базова інформація про ціни і дати.

Проте для сучасного користувача може не вистачати глибших фільтрів, візуальної привабливості та повної україномовної адаптації.

Наступний web-сайт, який ми розглянемо – це сайт турфірми «Coral Travel».

Посилання: https://www.coraltravel.ua/hot-offers/

Вигляд головної сторінки продемонстровано нижче на рисунку 1.5.

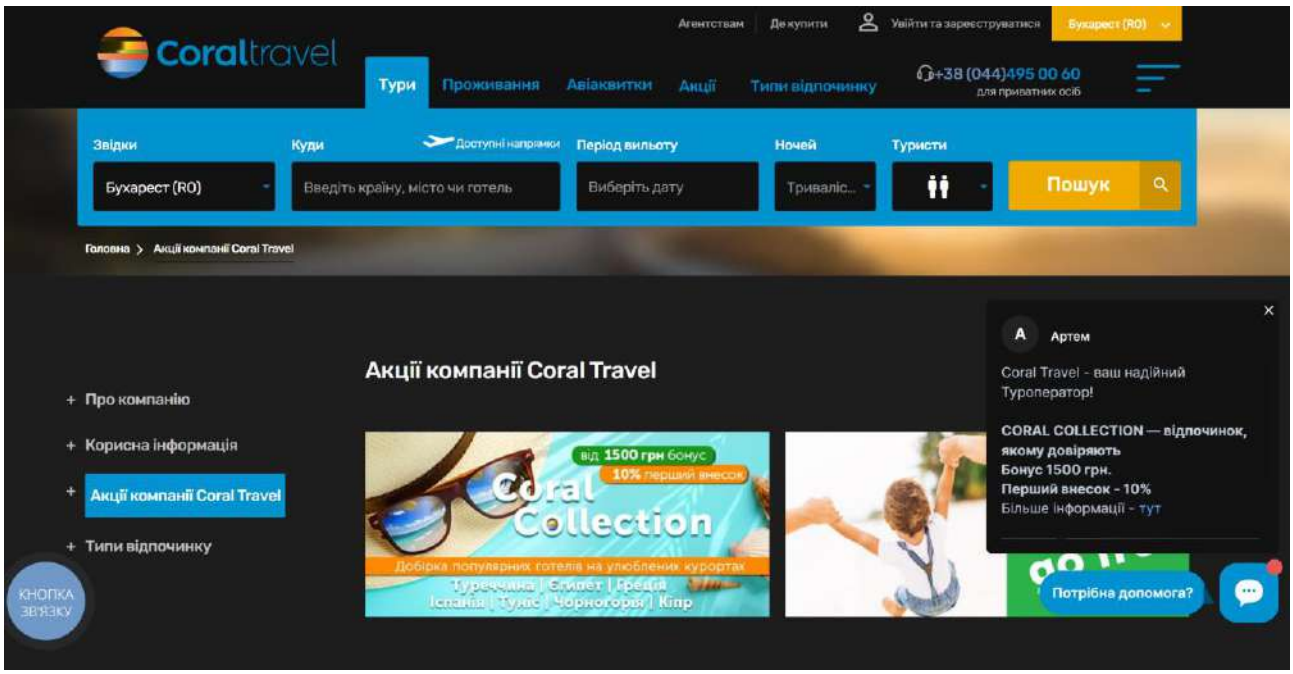


Рис. 1.5. Вигляд головної сторінки сайту турфірми «Coral Travel»

Серед переваг:

- Значні знижки на тури

  Гарячі тури пропонують знижки до 50% від стандартної ціни, що дозволяє заощадити без втрати якості обслуговування.

- Широкий вибір напрямків

Доступні популярні країни, такі як Туреччина, Єгипет, Греція, Іспанія, Таїланд, Мальдіви та інші.

- Зручний онлайн-пошук

  Інтерфейс дозволяє легко фільтрувати тури за країною, містом вильоту, датами, кількістю ночей та туристів.

- Додаткові сервіси

  Крім турів, сайт пропонує екскурсійні програми, круїзи та інші види відпочинку.

В ході аналізу окрім переваг, було виявлено такі недоліки:

- Обмежений вибір дат та напрямків

  Гарячі тури мають обмежені дати вильоту та напрямки, що може не відповідати вашим планам.

- Швидке прийняття рішень

  Необхідність швидко приймати рішення та готуватися до поїздки може бути стресовою для деяких туристів.

- Ризик відсутності місць

  Популярні пропозиції швидко розкуповуються, що може призвести до відсутності бажаного туру.

- Змішані відгуки про обслуговування

  Деякі користувачі скаржаться на проблеми з поверненням коштів та якістю обслуговування.

Висновок з огляду даного сайту можна зробити наступні.

Сайт Coral Travel пропонує вигідні гарячі тури з широким вибором напрямків та зручним онлайн-пошуком.

Проте, обмежений вибір дат, необхідність швидкого прийняття рішень та можливі проблеми з обслуговуванням можуть бути недоліками.

Рекомендується ретельно перевіряти умови туру та читати відгуки перед бронюванням.

1.3. Вхідна та вихідна інформація

Вхідна інформація

У даному програмному забезпеченні використовується основний вид вхідної інформації:

- дані про країни, міста та готелі, які включають географічне розташування, опис туристичних напрямків, назви готелів, ціни, умови проживання тощо

Уся інформація може бути взята з існуючої бази даних або додана вручну користувачем.

Вона зберігається у вигляді окремого файлу, що дозволяє легко імпортувати чи експортувати дані, а також відновити їх у разі потреби.

У цілому, обсяг вхідної інформації є невеликим і компактно представлений у вигляді зручної структури файлу бази даних.

Вихідна інформація

Вихідна інформація у програмному забезпеченні також поділяється на кілька типів:

- інформація про готелі та їхню вартість, яка відображається у зручному та візуально привабливому вигляді за допомогою HTML і OpenGL — із застосуванням текстур, градієнтів та інших графічних елементів; [4]
- вивід у форматі HTML із використанням скриптів для динамічного відображення даних, зокрема — побудова гістограм, які реагують на наведення курсору: при наведенні на готель користувач бачить фактичну вартість проживання за добу.
- експорт результатів у вигляді веб-сторінок або файлів для подальшого перегляду чи презентації;
- основна логіка роботи реалізована на мові програмування C#, яка відповідає за обробку інформації та передачу її до візуального інтерфейсу. [1]

Таким чином, результатом є зручна система для перегляду й аналізу туристичних пропозицій, що поєднує функціональність і наочність.

Висновки до розділу 1

У результаті проведеного аналізу було сформульовано основні вимоги до розроблюваного програмного забезпечення для організації туристичних турів.

Визначено основні функціональні й нефункціональні характеристики системи, такі як:

- можливість ведення бази даних турів і готелів,
- візуалізація статистичної інформації,
- експорт звітів у HTML-форматі,
- гнучке налаштування інтерфейсу користувача.

Під час огляду наявних програмних рішень виявлено, що більшість із них орієнтовані на великі туроператорські компанії, є складними у використанні або не підтримують необхідну візуалізацію та налаштування зовнішнього вигляду.

Це підтверджує доцільність розробки нового програмного продукту з урахуванням сучасних вимог до зручності, гнучкості та функціональності.

Було також проаналізовано вхідну та вихідну інформацію, яку система повинна обробляти: дані про тури, країни, готелі, ціни, періоди дії пропозицій тощо.

Це дозволило закласти основу для наступного етапу проектування системи та реалізації її програмних модулів..

РОЗДІЛ 2
РОЗРОБКА АЛГОРИТМУ РОЗВ’ЯЗАННЯ ЗАДАЧІ

2.1. Загальна характеристика задач

Процес розробки програмного забезпечення для організації туристичних турів потребує створення чіткого та ефективного алгоритму, що визначає логіку функціонування основних компонентів системи. Цей алгоритм має враховувати як збереження й обробку інформації про готелі та країни, так і формування HTML-звітів і побудову графічних візуалізацій.

Нижче наведено поетапний опис побудови алгоритмічного розв’язання задачі та його ключові характеристики.

1. Етапи побудови алгоритму
    - Формулювання мети

      На початковому етапі визначається основне завдання програми: автоматизація введення, обробки та виведення туристичної інформації з можливістю її візуалізації.

      Визначаються вхідні дані — назви готелів, країни, вартість, рейтинг тощо — та очікуваний результат: таблиці, HTML-звіти, графіки.
    - Формалізація вимог

      Алгоритм повинен забезпечувати ефективне оновлення даних, реагувати на дії користувача (додавання, редагування, видалення), дозволяти змінювати вигляд таблиці, шрифт і кольори.

      Передбачено також реалізацію побудови гістограм у вікні OpenGL на основі введених числових параметрів. [4]
    - Вибір логічних структур

      В алгоритмі використовуються умовні оператори для перевірки правильності введення даних, цикли для проходження масивів записів та обробки кожного готелю або туру, а також об'єктно-

орієнтовані підходи для роботи з формами, кнопками, таблицями та графічними об’єктами.

2. Основні характеристики алгоритму
   - Продуктивність

     Алгоритм має забезпечити мінімальний час відгуку при зміні або фільтрації інформації, швидке формування HTML-звіту, а також побудову 3D-графіків без затримок. Це досягається за рахунок ефективної реалізації сортування, обмеження глибини циклів і роботи з буферами у OpenGL. [4]
   - Гнучкість та розширюваність

     Структура алгоритму дозволяє додавати нові функціональні модулі (наприклад, підрахунок загальної вартості туру, сортування за рейтингом, фільтрацію по країні) без істотного перероблення існуючого коду.
   - Надійність

     Передбачено обробку помилок, пов’язаних з некоректним введенням даних або помилками читання файлів. Всі потенційно небезпечні дії контролюються через діалогові вікна з повідомленнями користувачу.
3. Оцінка складності
   - Базова складність

     Основні операції (додавання, пошук, побудова графіка) виконуються з лінійною або квазілінійною складністю (O(n) або O(n log n)), що є прийнятним для обсягу даних туристичної компанії малого чи середнього масштабу.
   - Оптимізація

     Застосовуються методи кешування даних, уникнення дублювання обробки візуальних елементів, а також динамічне оновлення тільки змінених частин інтерфейсу (наприклад, таблиць або окремих барів на гістограмі).

Це дозволяє зменшити споживання ресурсів та забезпечити плавну роботу додатку навіть при великій кількості записів.

2.2. Алгоритм виконання роботи

Основним фундаментом для розробки будь-якого програмного забезпечення є алгоритм – чітке та зрозуміле послідовне виконання завдань, спрямованих на досягнення конкретної мети або вирішення поставленої задачі.

Робота з програмою здійснюється на основі управління через пункти головного меню, які зображені нижче (Рис 2.1.).

Рис. 2.1. Вигляд головного меню програми

Було прийнято рішення створити чотири блок-схеми для детальної демонстрації алгоритму роботи програми. Щоб користувачу було просто і легко розібратись, блок-схеми створено зрозумілими і деталізованими.

Перша і початкова схема демонструє відкриття головного вікна програми і відображення пунктів меню (Рис. 2.2.):

- «Робота турфірми» та «Країни та міста» – дані пункти надають користувачу змогу працювати з інформацією БД;
- «3D» – відкриває вікно, на якому демонструється гістограма відображена в 3D-графіці (див. Додаток Б);
- «Налаштування» – користувач має змогу змінити оформлення зовнішнього виду таблиці та її вмісту;
- «Гістограма в HTML» – відкриває HTML-сторінку, на якій гістограма відображається у 2D-графіці (див. Додаток А);
- «Допомога» – пояснює користування даним додатком, в якому міститься інформація з поясненнями;
- «Про програму» – даний пункт містить інформацію про автора (див. Додаток В).

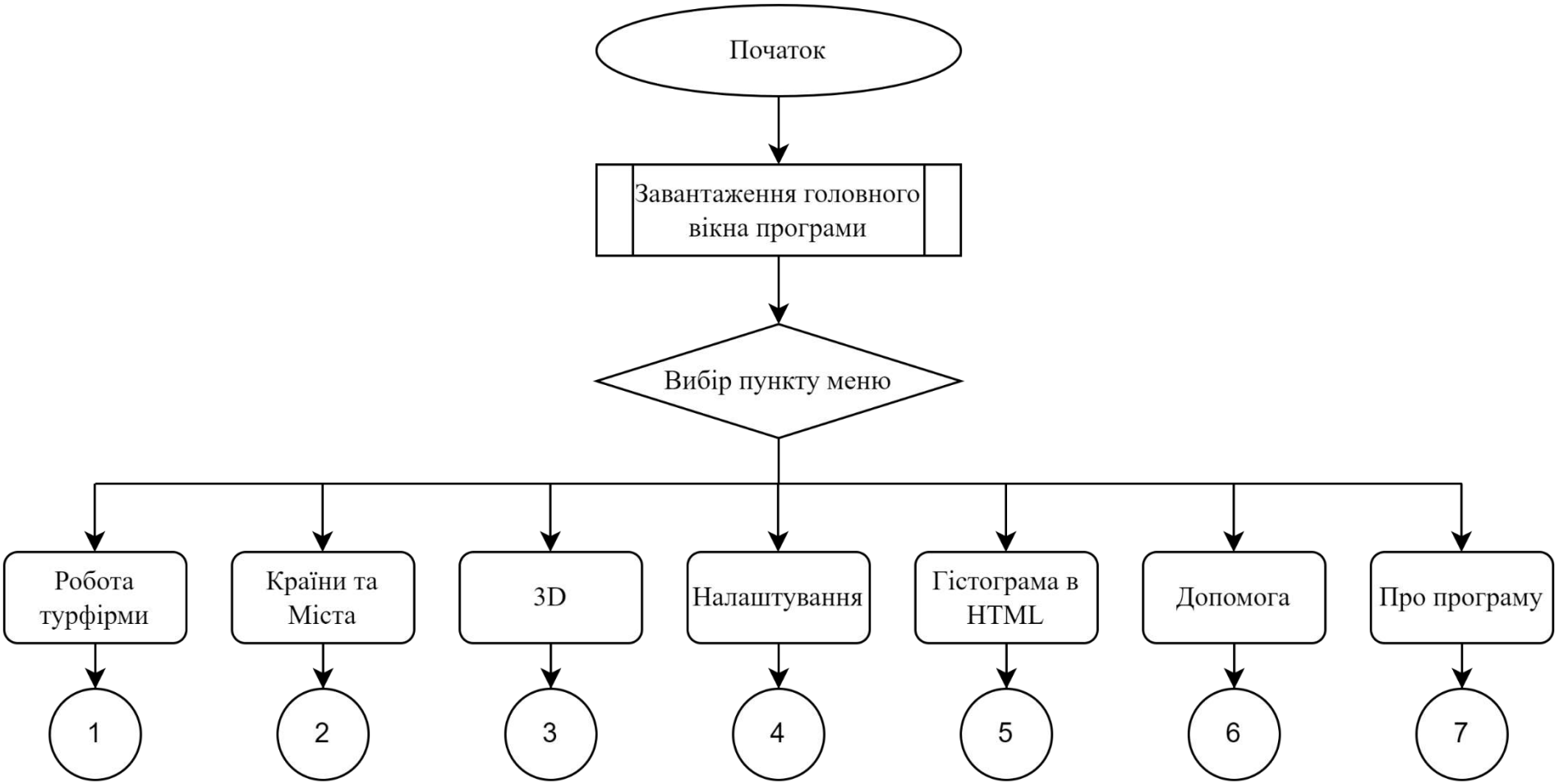


Рис. 2.2. Блок-схема початку роботи з програмою

Далі розглянемо пункт головного меню «Робота турфірми», який є одним з центральних у програмі.

Його логіка зображена на блок-схемі (Рис. 2.3). Після вибору цього пункту відкривається випадаюче меню з чотирма підпунктами: «Інформація по туристам», «Готелі», «Доступні готелі», «Вартість туру». Кожен з них виконує окрему функцію та має власний обробник події.

Даний пункт меню має випадаюче меню, містить в собі чотири пункти:

- «Інформація по туристам» – даний пункт відображає інформацію в табличному вигляді за чотирма критеріями:
  - «Клієнти» – містить в собі інформацію про клієнта: ПІБ, дата народження, віза та зарплатня;
  - «Готелі» – дозволяє переглянути повні характеристики обраного об’єкта: місто, назва, клас, місця, екскурсії, харчування, вартість та наявність басейну та ресторану;
  - «Договори на тури» – містить в собі інформацію про договори: дати початку та завершення туру, країну та місто, вартість, наявність знижки, етап виконання, вартість білетів, тим номеру та доповнення;
  - «До сплати клієнту» – містить в собі інформацію, яка відображає загальну вартість за тур.
- «Готелі» – даний пункт дає користувачу обрати країну, місто та готель, і переглянути повну інформацію про обраний готель;
- «Доступні готелі» – даний пункт дає змогу обрати готель та дати і переглянути чи будуть виконані всі вимоги користувача та перевірити відповідність параметрам;
- «Вартість туру» – даний пункт надає змогу користувачу прорахувати та забронювати необхідний тур.

Така структура дозволяє реалізувати гнучку логіку навігації й забезпечити швидкий доступ до ключової інформації.

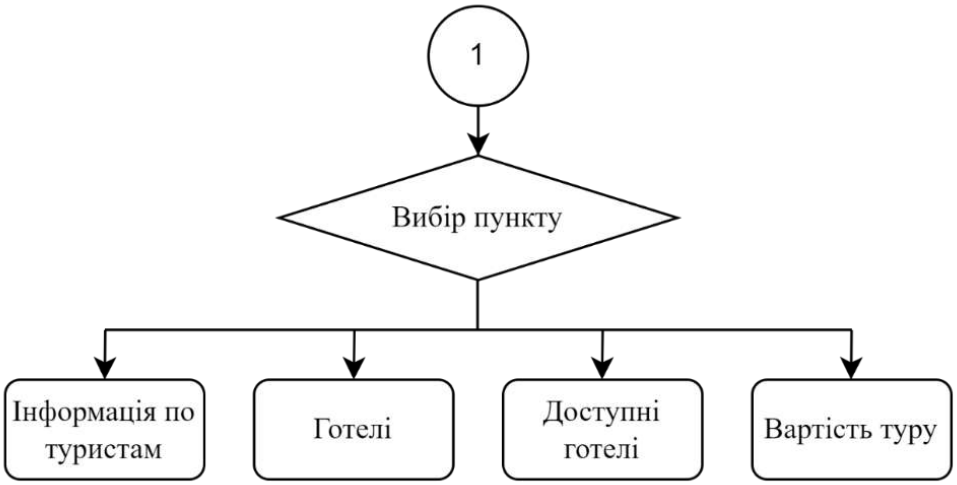


Рис. 2.3. Блок-схема пункту «Робота турфірми»

Наступний пункт, який ми розглянемо – це «Країни та міста». Схематично зображено нижче (Рис. 2.4.).

Даний пункт також має випадаюче меню, воно має два пункти:

- «Країни» – даний пункт дає змогу додати, редагувати та видалити країну з БД;
- «Міста» – даний пункт також надає змогу додати, редагувати та видалити місто з БД, тільки місто додається при виборі країни.

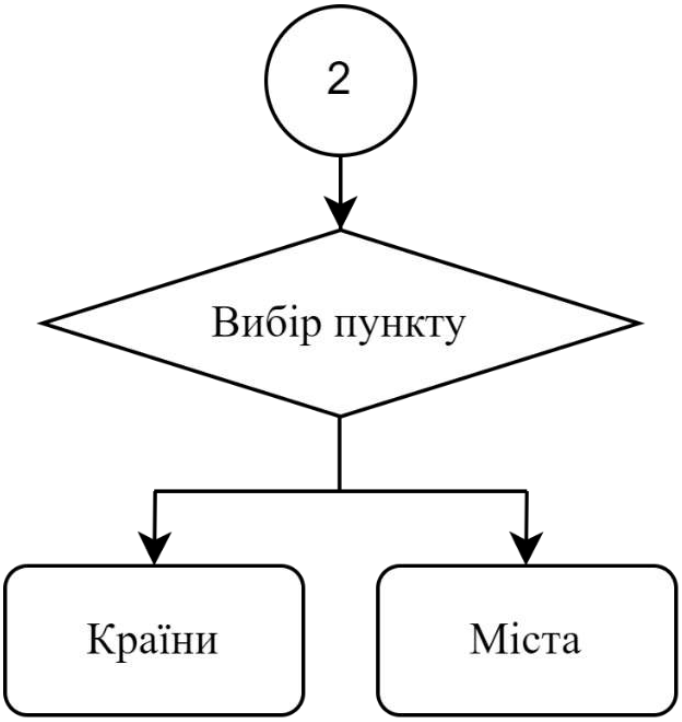


Рис. 2.4. Блок-схема пункту «Країни та міста»

Далі розглянемо пункт головного меню «Налаштування». Його схематично відображено на рисунку 2.5.

При настисканні на даний пункт меню відкривається випадаюче меню, яке містить в собі наступні підпункти:

- «Налаштування шрифту» – даний пункт відкриває діалогове вікно для вибору характеристик шрифту;
- Пункти «Налаштування кольору заголовку», «Налаштування кольору заголовків стовпців» та «Налаштування кольору тексту таблиці» – мають один принцип роботи, при виборі одного з цих пунктів відкривається діалогове вікно з вибору кольорової палітри.

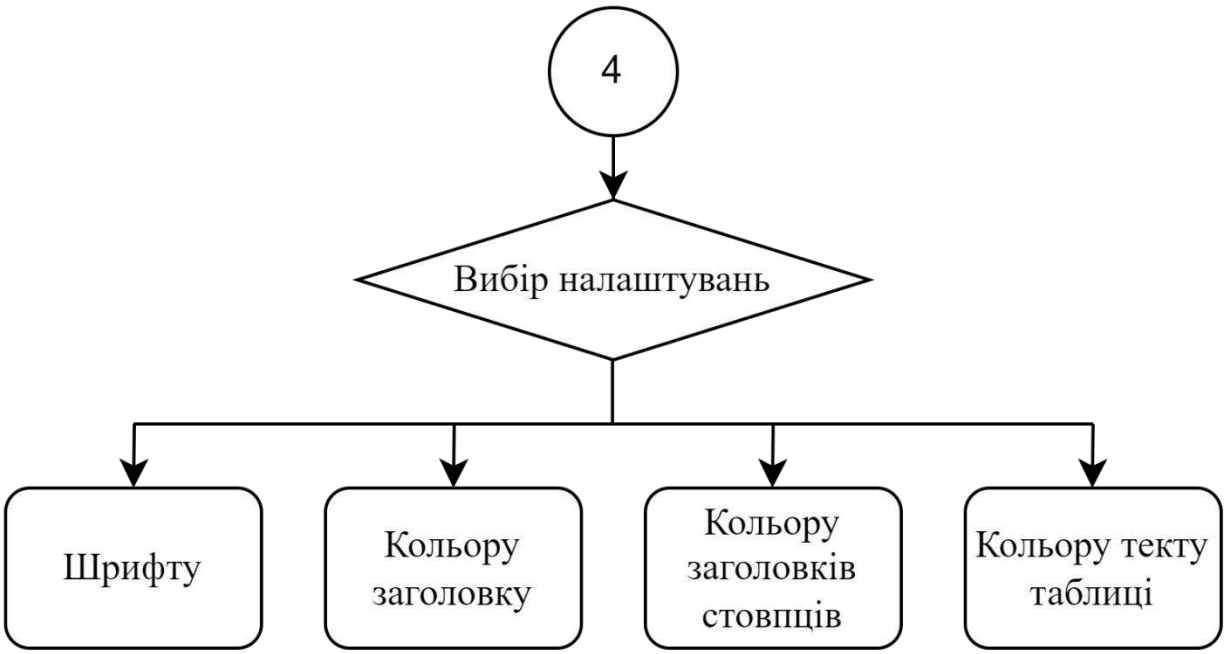


Рис. 2.5. Блок-схема пункту «Країни та міста»

Робота з програмою припиняється тоді, коли користувач задовольнить свої потреби.

Висновки до розділу 2

У цьому розділі розглянуто алгоритм функціонування програмного забезпечення для організації туристичних турів.

Основну увагу приділено покроковому проектуванню логіки роботи програми, зокрема:

- обробці даних про готелі,
- їх відображенню в таблиці,
- налаштуванню зовнішнього вигляду,
- побудові 3D-графіків,
- формуванню HTML-звітів.

Алгоритм передбачає взаємодію користувача з головним вікном через меню та елементи керування, обробку подій натискання, динамічне оновлення таблиць, виклик стандартних діалогів для зміни шрифтів і кольорів.

Окрема частина алгоритму відповідає за візуалізацію даних засобами OpenGL — створення гістограми, ініціалізацію буферів, відображення об'єктів сцени.

Завдяки чіткій структурі та оптимізації, алгоритм забезпечує стабільну, гнучку та ефективну роботу програми, дозволяє швидко оновлювати інформацію, змінювати налаштування та формувати наочне графічне представлення туристичних даних.

РОЗДІЛ 3
ОРГАНІЗАЦІЯ ІНФОРМАЦІЙНОГО ЗАБЕЗПЕЧЕННЯ

3.1. Загальна характеристика інформаційного забезпечення

Інформаційне забезпечення є необхідним елементом в сучасних умовах розвитку технологій та обробки даних.

Це обумовлено сукупністю документів, нормативної бази та прийнятих рішень, які визначають обсяг, розміщення та форми організації інформації в системі.

Принципи створення інформаційного забезпечення визначаються основними цілями та вимогами до його функціонування.

Серед них варто відзначити:

- цілісність,
- достовірність,
- контроль,
- захист від несанкціонованого доступу,
- єдність і гнучкість,
- стандартизацію та уніфікацію,
- адаптивність та мінімізацію помилок введення-виведення інформації.

Розробка інформаційного забезпечення визнається однією з ключових складових інформаційної системи.

Вона націлена на досягнення єдності та збереження інформації, необхідної для розв'язання завдань, а також на створення єдиної структури для всіх завдань інформаційних систем. [9]

Об’єкт та предмет дослідження визначаються як оптимізаційні задачі, що виникають у різних галузях.

Застосування розробленого програмного забезпечення на мові C# спрямоване на підвищення ефективності організації туристичних турів,

автоматизацію обліку клієнтів та зменшення часових витрат при плануванні маршрутів. [1]

Інформаційна база, як основна складова інформаційного забезпечення системи організації туристичних турів, поділяється на внутрішньо-машинну та зовнішньо-машинну.

Внутрішньо-машинна інформаційна база реалізована у вигляді структурованих таблиць бази даних, які містять оперативну інформацію про клієнтів, тури, бронювання та нормативно-довідкові дані (наприклад, ціни, маршрути, умови проживання).

Зовнішньо-машинна база включає документи, договори, туристичні путівки та інші матеріали, призначені для безпосередньої роботи співробітників та клієнтів.

Для забезпечення гнучкості та ефективного управління даними в системі організації туристичних турів використовується реляційна база даних з чіткою структурою таблиць.

Розроблена система класифікації та кодування туристичних напрямків, клієнтів і послуг, а також уніфіковані форми документообігу підвищують ефективність роботи туроператора та забезпечують наукову цінність дослідження.

Інформаційне забезпечення системи організації туристичних турів включає методичні матеріали, систему класифікації та кодування туристичних напрямків, а також структуровану інформаційну базу даних.

Внутрішньо-машинна база даних забезпечує ефективне управління інформацією про клієнтів, тури та бронювання, тоді як зовнішньо-машинна база включає документообіг та звітні форми, необхідні для повноцінної роботи туристичного агентства.

Загалом, інформаційне забезпечення є ключовим елементом для ефективного функціонування системи організації туристичних турів, забезпечуючи автоматизацію обліку клієнтів, управління туристичними пакетами та аналіз попиту.

База даних туристичної системи представляє собою структуровану сукупність взаємопов'язаних даних про клієнтів, тури, бронювання та супутні послуги, організовану за принципами цілісності, мінімальної надмірності та ефективного доступу до інформації, що відповідає специфіці туроператорської діяльності.

Предметна область дослідження охоплює інформаційні процеси туристичного агентства, зокрема управління клієнтською базою, формування турів, бронювання послуг та аналіз ринкового попиту, що підлягають автоматизації шляхом створення спеціалізованої бази даних.

Моделі даних у системі організації туристичних турів визначають оптимальну логічну структуру для зберігання інформації, забезпечуючи ефективні зв'язки між сутностями (клієнти, тури, послуги) та підтримку цілісності даних у процесі експлуатації системи.

База даних туристичної системи - це структурований цифровий архів, що містить взаємопов'язані дані про клієнтів, туристичні пакети, бронювання та супутні послуги, організовані для ефективного пошуку та обробки.

- Функціональність СУБД у туроператорській системі включає: Визначення даних - формалізацію структури інформації про тури, клієнтів та угоди;
- Обробку даних - оперативне оновлення інформації про наявність місць, ціноутворення;
- Управління даними - контроль цілісності та безпеки турбази.

  Реляційна модель реалізована через систему взаємозв'язаних таблиць (клієнти, тури, замовлення), що забезпечує гнучкість у формуванні турпакетів та генерації звітності.

Логічна структура реляційної таблиці визначається згідно з реквізитним складом об'єктів туристичної сфери, що включає унікальне ім'я таблиці, склад атрибутів, тип даних, властивості даних та ключі.

Бази даних туристичної системи потребують особливого контролю та захисту, що лежить на адміністраторі бази даних, з огляду на конфіденційність клієнтської інформації.

Створення бази даних за допомогою СУБД MS Access, реляційної системи управління базами даних, забезпечує необхідну структурованість, наявність ключових полів та зв'язків між даними про тури, клієнтів та бронювання.

Проектування інформаційної системи та БД для розробленого програмного продукту організації туристичних турів відповідає всім вимогам стосовно структурованості, ключових полів, захисту та обмеження доступу до даних.

3.2. Побудова системи класифікації та кодування

Класифікація об'єктів у системі організації туристичних турів може здійснюватися за допомогою ієрархічного або фасетного методів.

- Ієрархічний метод.

  Передбачає послідовний поділ об'єктів на підпорядковані групи, що дозволяє чітко структурувати інформацію в системі.

- Фасетний метод

  Дає змогу класифікувати туристичні пропозиції за різними незалежними характеристиками, що забезпечує гнучкість у роботі з даними. Обидва методи сприяють ефективній організації інформації в базі даних.

Використання системи класифікації та кодування у програмному забезпеченні для туристичних турів дозволяє оптимізувати процеси обробки даних та покращити якість обслуговування клієнтів.

Кодування є важливим етапом структурування інформації про туристичні послуги, що передбачає створення унікальних кодів для класифікованих об'єктів системи.

В туристичній сфері коди повинні ефективно відображати логічну структуру даних, забезпечувати точну ідентифікацію турів, готелів, маршрутів, а також чітко визначати взаємозв'язки між ними.

У практиці організації туристичних турів застосовуються різні методи кодування, які поділяються на дві основні групи: реєстраційні та класифікаційні.

До реєстраційних методів, які відрізняються простотою та однозначністю, належать порядковий і серійно-порядковий методи.

Порядковий метод передбачає послідовне присвоєння номерів з натурального ряду, тоді як серійно-порядковий метод дозволяє закріплювати певні серії номерів за об'єктами зі схожими характеристиками.

Для туристичних систем особливе значення має послідовний метод кодування, який ґрунтується на принципах ієрархічної класифікації.

Цей метод дозволяє створювати логічні кодові послідовності, що відповідають вкладеним рівням структури туристичних пропозицій, забезпечуючи при цьому зручність обробки та аналізу даних.

Паралельний метод кодування знаходить активне застосування в системі організації туристичних турів, де він використовує коди незалежних категорій, сформованих на основі фасетного методу класифікації.

Цей підхід особливо ефективний для роботи з різнорідними характеристиками турів, такими як типи проживання, види харчування, категорії транспорту та інші параметри, що дозволяє гнучко керувати туристичними продуктами.

Результати процесів класифікації та кодування в системі організації турів знаходять своє відображення у спеціальних нормативних документах - класифікаторах.

Туристичний класифікатор представляє собою строго систематизований перелік усіх об'єктів класифікації (турів, послуг, напрямків) з їх точними назвами та відповідними кодами.

Такі документи є основою для стандартизації туристичних продуктів та послуг.

Розробка сучасних інформаційних систем для туристичного бізнесу, зокрема міжнародного, вимагає комплексного підходу до проектування різних видів забезпечення. [9]

Інформаційне забезпечення в цьому контексті грає ключову роль, оскільки саме воно забезпечує ефективне функціонування всіх компонентів системи організації туристичних турів, від ведення клієнтської бази до управління туроперациями.

Інформаційна база в системі автоматизації туристичної діяльності формується з урахуванням специфічних бізнес-процесів та економічних показників, які підлягають автоматизації.

Особливий акцент робиться на розробці ефективних систем класифікації та кодування, що забезпечують стандартизацію та узгодженість усіх інформаційних потоків у сфері туристичних послуг.

Процеси класифікації та кодування туристичних продуктів офіційно закріплюються у класифікаторах, які набувають статусу нормативно-технічних документів.

У міжнародній практиці існують уніфіковані класифікаційні системи, що слугують зразком для розробки відповідних стандартів у туристичній галузі.

Сучасні туристичні системи використовують різноманітні підходи до кодування інформації:

- порядкові та серійні системи для послідовної ідентифікації;
- десяткові схеми для структурованого поділу категорій;
- шахові методи для відображення комплексних взаємозв'язків;
- комбіновані підходи, що поєднують різні методи кодування.

Ці системи базуються на фундаментальних принципах класифікації, включаючи порядковий, послідовний та фасетний методи, що дозволяє оптимально організувати інформаційні ресурси туристичного бізнесу.

Порядкова система кодування передбачає послідовне присвоєння ідентифікаторів, що спрощує облік, але обмежує гнучкість при розширенні класифікації.

Серійна система вирішує цю проблему через виділення діапазонів кодів для окремих категорій, забезпечуючи структурований підхід до класифікації, проте вимагає чіткого планування серій.

Десяткова система ґрунтується на ієрархічній структурі кодів, що дозволяє логічно групувати дані, але може збільшувати довжину ідентифікаторів.

Шахова система застосовується для відображення комплексних взаємозв’язків між параметрами, що особливо актуально для динамічних систем управління.

Системи повторення та комбіновані методи кодування застосовуються в залежності від специфіки туристичних продуктів, при цьому їх ефективність визначається конкретними умовами використання.

Операція перекодування передбачає присвоєння нових ідентифікаторів елементам туристичних каталогів.

Для цього можуть використовуватися спеціальні таблиці відповідності, що забезпечують узгодженість між різними системами класифікації.

Інформаційно-довідкова підсистема включає інтерфейс для управління даними, який дозволяє здійснювати модифікацію та збереження інформації в базі даних.

Модуль планування турів є технічно складним компонентом, оскільки повинен забезпечувати автоматичне формування маршрутів з урахуванням множини параметрів.

Складова управління турпродуктами надає можливості визначення характеристик для кожного елементу туристичної програми.

Архітектура системи передбачає чітку ієрархію: базовий рівень взаємодії з даними реалізований через модуль роботи з БД, на який спираються всі інші функціональні компоненти.

Ця структура забезпечує гнучкість у керуванні туристичними ресурсами та ефективне функціонування всієї системи в цілому.

3.3. Характеристика та вимоги до бази даних.

База даних системи організації туристичних турів являє собою структуроване сховище інформації, призначене для ефективного управління всіма даними, необхідними для роботи туристичного агентства.

Вона забезпечує надійне зберігання, швидкий доступ та цілісність інформації про тури, клієнтів, маршрути, готелі та інші пов'язані сутності. Основна функція бази даних полягає у систематизації великих обсягів інформації та забезпеченні їхньої узгодженості.

Структура бази даних включає два основних компоненти: безпосередньо сховище даних у вигляді взаємопов'язаних таблиць та механізми управління цими даними.

Таблиці організовані таким чином, щоб відображати ключові сутності туристичного бізнесу та зв'язки між ними.

До бази даних туристичної системи висуваються суттєві вимоги щодо нормалізації, що дозволяє уникнути дублювання інформації та забезпечити її цілісність.

Важливим аспектом є продуктивність роботи з даними, особливо при виконанні складних запитів, пов'язаних з пошуком та фільтрацією турів. Система повинна забезпечувати безпеку даних через механізми авторизації та регулярне резервне копіювання.

Архітектура бази даних має бути достатньо гнучкою, щоб дозволяти розширення функціоналу без необхідності значних змін у структурі.

Реалізація бази даних заснована на реляційній моделі, де інформація організується у вигляді таблиць з чітко визначеними зв'язками між ними.

Такий підхід дозволяє ефективно керувати складними взаємозв'язками між різними аспектами туристичної діяльності та забезпечує стабільну роботу всієї системи організації турів.

Система управління базами даних (СУБД) для туристичної системи є ключовим програмним комплексом, який забезпечує створення, обслуговування та ефективне використання бази даних.

СУБД надає інструменти для оновлення інформації, забезпечення цілісності даних, захисту від несанкціонованого доступу та формування результатів у відповідь на користувацькі запити.

Фундаментом організації даних у туристичній системі служить модель даних – сукупність принципів та правил представлення інформації.

В історичній перспективі розрізняють три основні підходи до структурування даних:

- Ієрархічна модель організує інформацію у вигляді деревоподібної структури, де кожен елемент може мати лише одного предка.

  Такий підхід може бути корисним для представлення певних аспектів туристичних продуктів.
- Мережна модель розширює можливості ієрархічної, дозволяючи елементам мати кілька зв'язків, що більш точно відображає складні взаємозв'язки в туристичній галузі.
- Реляційна модель, яка є основною для сучасних туристичних систем, організовує дані у вигляді таблиць з чітко визначеними зв'язками між ними.

  Цей підхід забезпечує гнучкість та ефективність при роботі з великими обсягами інформації про тури, клієнтів та ресурси.

Сучасні тенденції вказують на зростання популярності об'єктно-орієнтованих підходів до організації даних, які дозволяють більш точно моделювати складні туристичні об'єкти та їх взаємодії.

Однак реляційна модель залишається основним вибором для більшості комерційних систем організації туристичних турів через свою надійність та доведену ефективність.

У контексті розробки системи організації туристичних турів особливе значення набувають різні моделі організації даних, кожна з яких має свої особливості:

- Ієрархічна модель організує дані у вигляді деревоподібної структури, де кожен дочірній елемент має лише одного батьківського.

  Такий підхід може бути корисним для представлення структурованих туристичних продуктів, де чітко визначені відносини підпорядкування.
- Мережна модель розширює можливості ієрархічної, дозволяючи елементам мати множинні зв'язки.

  Це особливо актуально для туристичної сфери, де один і той же ресурс (наприклад, готель) може входити до різних турів або маршрутів.
- Реляційна модель, як найпоширеніша у сучасних туристичних системах, представляє дані у вигляді таблиць з чіткими зв'язками між ними.

  Такий підхід забезпечує просту та ефективну роботу з інформацією про клієнтів, тури, бронювання та інші ключові компоненти системи.

При проектуванні бази даних для туристичної системи дотримуються таких ключових принципів:

- забезпечення високої продуктивності при обробці запитів;
- можливість легкого оновлення та зміни даних;
- незалежність зберігаємої інформації від конкретної реалізації;
- підтримка багатокористувацького режиму роботи;
- реалізація надійних механізмів захисту даних;
- відповідність стандартам у сфері туристичних послуг;
- точне відображення всіх аспектів туристичної діяльності.

Ці принципи забезпечують створення ефективної та надійної бази даних, здатної повноцінно підтримувати всі функції системи організації туристичних турів.

Для інформаційного забезпечення системи організації туристичних турів висуваються такі ключові вимоги.

По-перше, інформаційна база повинна бути достатньо повною для автоматизації всіх функцій системи, включаючи управління турами, клієнтами та бронюванням.

Важливим аспектом є використання узгоджених систем кодування - власних класифікаторів для внутрішніх даних та стандартизованих кодів для зовнішнього обміну інформацією.

Система має забезпечувати сумісність з іншими інформаційними системами через уніфікацію формату даних, методів адресації та способів подання інформації.

Документообіг повинен відповідати вимогам УСД та внутрішнім стандартам компанії, а форми введення/виведення даних - технічним характеристикам використовуваного обладнання.

Особливу увагу приділяється узгодженості форм вихідної інформації з потребами користувачів, використанню загальноприйнятої термінології та скорочень.

Система повинна включати механізми контролю актуальності даних, їх оновлення та відновлення після можливих збоїв.

Для реалізації бази даних обрано Microsoft Access, який забезпечує необхідний функціонал через:

- інтуїтивний інструментарій для створення таблиць;
- можливість розробки зручних екранних форм;
- підтримку SQL-запитів різного рівня складності;
- генерацію звітів для подальшого друку;
- ефективні засоби експорту та імпорту даних.

Цей підхід дозволяє створити надійну інформаційну основу, здатну задовольнити всі вимоги сучасного туристичного бізнесу.

3.4. Структура бази даних.

База даних системи організації туристичних турів реалізована у форматі *mdb з використанням Microsoft Access як основної СУБД.

Цей вибір обґрунтований наступними ключовими перевагами даної платформи:

1. Інтуїтивно зрозумілий графічний інтерфейс, який значно спрощує процес створення та адміністрування бази даних, особливо для користувачів без глибоких технічних знань.
2. Компактна структура зберігання - усі дані системи (таблиці, зв'язки, форми, звіти) інтегровані в єдиний файл, що забезпечує зручність управління та перенесення.
3. Наявність вбудованих майстрів, які автоматизують створення основних елементів системи та спрощують виконання типових операцій.
4. Широка підтримка з боку Microsoft, включаючи регулярні оновлення та гармонійну інтеграцію з іншими продуктами компанії.
5. Оптимізована робота в середовищі Windows, що гарантує стабільність та швидкодію.
6. Потужні інструменти імпорту/експорту даних, що дозволяє працювати з різними форматами (Excel, текстові файли) та інтегруватися з іншими СУБД через ODBC.
7. Розширені можливості кастомізації за рахунок використання VBA та макрокоманд для автоматизації складних процесів.
8. Гнучкі засоби розробки, що дозволяють створювати спеціалізовані рішення для туристичного бізнесу.

Такий підхід забезпечує оптимальний баланс між функціональністю, надійністю та зручністю використання, що є критично важливим для ефективної роботи системи організації туристичних турів.

У подальшому аналізі буде детально розглянуто конкретну реалізацію структури даних та їх взаємозв'язків.

Структура бази даних включає об'єкти головної програми, такі як таблиці – основні довідники для створення документів та запити – самі документи, що формуються із даних довідників.

Дана база даних має назву «турист.mdb» і містить в собі таблиці:

- «Dogovor» – ця таблиця містить інформацію про договори на туристичні послуги, зокрема дати початку та завершення турів, а також додаткові параметри;
- «Goroda» - ця таблиця є довідником міст, які пов’язані з туристичними послугами.

  Вона містить інформацію про кожне місто, його унікальний ідентифікатор та ідентифікатор країни, до якої воно належить;
- «Klienti» - ця таблиця містить інформацію про клієнтів туристичної агенції, включаючи їхні персональні дані, фінансову інформацію та дані про візи;
- «Klienti_po_dogovory» - ця таблиця виконує функцію зв'язку між договорами та клієнтами, вказуючи, які клієнти прив'язані до яких договорів.

  Вона відображає відношення "багато до багато" (один клієнт може мати кілька договорів, один договір може належати кільком клієнтам);
- «Otel» - ця таблиця містить детальну інформацію про готельні об'єкти, включаючи їх характеристики, послуги, вартість проживання та інші параметри, що стосуються комфорту та інфраструктури;

- «Strana» - ця таблиця містить про країни, з якими пов’язані готелі або туристичні напрямки.

  Вона використовується для ідентифікації країн за їхніми кодами та назвами;

- «Tip_nomera» - ця таблиця містить типи номерів у готелях, їхні позначення та додаткові характеристики (наприклад, доплату за покращений номер).

  Використовується для класифікації номерів та розрахунків вартості проживання;

- «Tyri» - ця таблиця містить туристичні тури, включаючи їх вартість, знижки, статус виконання, валюту оплати, а також пов'язані з ними готелі, міста та типи номерів.

  Використовується для керування туристичними пакетами та аналізу продажів.

Таблиця «Dogovor» складається з 4 полів, які містять в собі наступну інформацію (Рис. 3.1.):

- Унікальний номер договору
- Дата початку туру
- Номер туру
- Дата завершення туру

Dogovor

Имя поля	Тип данных
Nom_dogovora	Счетчик
Data_nachala_tyra	Дата и время
Tyr	Числовой
Data_okonch_tyra	Дата и время

Рис. 3.1. Таблиця «Dogovor»

У таблиці «Dogovor» – містяться дані початку/закінчення дії договору та ідентифікатор туру.

Вигляд таблиці наведено на рисунку 3.2.

Dogovor

Nom_dogovora	Data_nachala_tyra	Tyr	Data_okonch_tyra
16	28.03.2009	38	31.03.2009
17	11.04.2009	40	16.04.2009
18	03.04.2009	44	09.04.2009
19	19.04.2009	45	26.04.2009

Рис. 3.2. Дані початку/закінчення дії договору та ідентифікатор туру

Опис властивостей стовпчиків таблиці «Dogovor» наведено у табл. 3.1.

Таблиця 3.1

Опис властивостей стовпчиків таблиці «Dogovor»

Найменування	Поле	Тип поля	Довжина	Первинний ключ	Обов'язкове поле	Індексне поле
Унікальний номер договору	Nom_dogovora	Лічильник	Довге ціле	+	Так	ІНД
Дата початку туру	Data_nachala_tyra	Дата/час	Короткий формат дати	-	Так	-
Номер туру	Tyr	Числове	Довге ціле	-	Ні	-
Дата завершення туру	Data_okonch_tyra	Дата/час	Короткий формат дати	-	Так	-

У таблиці «Goroda» вміщуються дані назв міст та ідентифікатор країни (Рис. 3.3.).

Goroda

Kod	Kod_str	Gorod
8	2	Аттика (Афіни)
9	2	Острів Крит
10	2	Кіклади
11	2	Керкіра (Корфу)
12	2	Острови Додеканісу
13	2	Родос

Рис. 3.3. Дані таблиці «Goroda»

Структура таблиці «Goroda» у системі керування базами даних Microsoft Access зображено на рисунку 3.4.

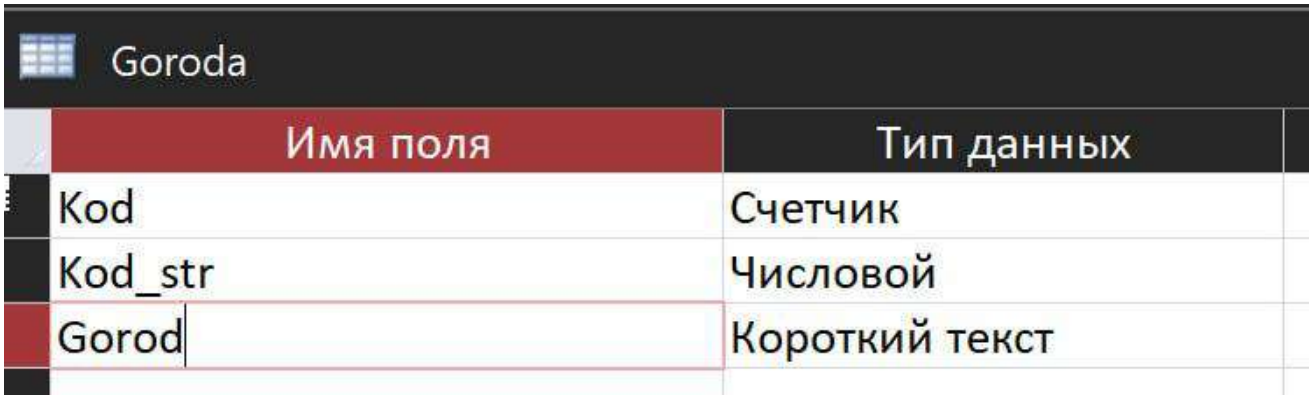


Рис. 3.4. Структура таблиці «Goroda»

Структура таблиці «Goroda» складається з 3 полів:

- Унікальний ідентифікатор міста;
- Ідентифікатор країни;
- Назва міста.

Кожне поле є невід'ємною частиною таблиці. Детальний опис структури подано у вигляді таблиці.

Представлено нижче у табл. 3.2.

Таблиця 3.2

Структура таблиці «Goroda»

Найменування	Поле	Тип поля	Довжина	Первинний ключ	Обов'язкове поле	Індексне поле
Унікальний ідентифікатор міста	Kod	Лічильник	Довге ціле	+	Так	ІНД
Ідентифікатор країни	Kod_str	Ціле число	Довге ціле	-	Ні	-
Назва міста	Gorod	Текстове	50	-	Ні	-

Таблиця «Klienti» складається з 7 полів, які містять в собі наступну інформацію (Рис. 3.5.):

- Унікальний ідентифікатор клієнта
- Прізвище клієнта
- Ім’я клієнта
- Номер візи
- Дата народження клієнта
- Зарплата клієнта
- Наявність страховки

Klienti

Имя поля	Тип данных
Kod_klienta	Счетчик
Familija	Короткий текст
Imia	Короткий текст
Viza	Короткий текст
God_rogd	Дата и время
Zarplata	Числовой
Strahovka	Логический

Рис. 3.5. Таблиця «Klienti»

У таблиці «Klienti» – містяться дані про клієнта компанії.

Вигляд таблиці наведено на рисунку 3.6.

Klienti

Kod_klienta	Familija	Imia	Viza	God_rogd	Zarplata	Strahovka
0001	Іваненко	Євген	2355656	14.02.1981	3455	☐
0002	Ткач	Микола	4564778	23.06.1973	5367	☐
0003	Кравець	Світлана	7867888	17.05.1983	587	☐
0004	Козак	Михайло	9789556	13.02.1975	5785	☐
0005	Клименко	Денис	6786786	18.11.1971	5645	☐
0006	Сидоренко	Станіслав	6787699	13.04.1972	3467	☐
0007	Марченко	Аркадій	7886785	29.01.1979	9077	☐
0008	Руденко	Анжела	5276789	14.01.1965	3478	☐
0009	Павленко	Сергій	7786788	13.02.1975	7899	☐
0010	Кузьменко	Ірина	6786799	11.09.1963	1234	☐

Рис. 3.6. Дані клієнтів компанії

Опис властивостей стовпчиків таблиці «Klienti» наведено у табл. 3.3.

Таблиця 3.3

Опис властивостей стовпчиків таблиці «Klienti»

Найменування	Поле	Тип поля	Довжина	Первинний ключ	Обов'язкове поле	Індексне поле
Унікальний ідентифікатор клієнта	Kod_klienta	Лічильник	Довге ціле	+	Так	ІНД
Прізвище клієнта	Familija	Текстове	50	-	Так	Так
Ім'я клієнта	Imia	Текстове	30	-	Так	-
Номер візи	Viza	Текстове	50	-	Ні	-
Дата народження клієнта	God_rogd	Дата/час	Короткий формат дати	-	Так	-
Зарплата клієнта	Zarplata	Ціле число	Довге ціле	-	Ні	-
Наявність страховки	Strahovka	Логічний	1	-	Так	-

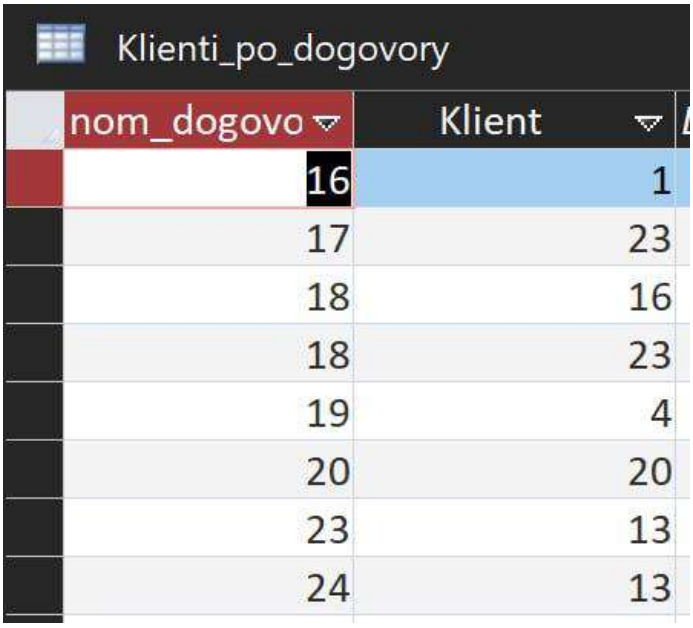
Таблиця «Klienti_po_dogovory» складається з 2 полів, які містять в собі наступну інформацію (Рис. 3.7.):

- Номер договору
- Код клієнта

Рис. 3.7. Таблиця «Klienti_po_dogovory»

У таблиці «Klienti_po_dogovory» – містяться дані для зв'язку таблиць.

Вигляд таблиці наведено на рисунку 3.8.



Klienti_po_dogovory

nom_dogovo	Klient
16	1
17	23
18	16
18	23
19	4
20	20
23	13
24	13

Рис. 3.8. Дані для зв’язку таблиць

Опис властивостей стовпчиків таблиці «Klienti_po_dogovory» наведено у табл. 3.4.

Таблиця 3.4

Опис властивостей стовпчиків таблиці «Klienti_po_dogovory»

Найменування	Поле	Тип поля	Довжина	Первинний ключ	Обов’язкове поле	Індексне поле
Номер договору	nom_dogovor	Ціле число	Довге ціле	+	Так	+
Код клієнта	Klient	Ціле число	Довге ціле	+	Так	+

Таблиця «Otel» складається з 24 полів, які містять в собі наступну інформацію (Рис. 3.9. та Рис. 3.10):

- Унікальний ідентифікатор готелю
- Код міста
- Назва готелю
- Кількість зірок готелю
- Кількість місць у готелі

- Кількість екскурсій
- Тип харчування (кількість разів)
- Вартість проживання за добу
- Наявність власної берегової лінії
- Наявність басейну
- Наявність ресторану
- Зображення або інший OLE-об'єкт
- Номер карти
- Наявність мінібару
- Наявність телевізора
- Наявність інтернету
- Наявність фітнес-залу
- Наявність обміну валюти
- Наявність дайвінг-центру
- Наявність прибирання номеру
- Наявність сауни
- Наявність бізнес-залу
- Наявність SPA-центру
- Розмір знижки

Otel

Имя поля	Тип данных
Kod_otel	Счетчик
Kod_gor	Числовой
Name	Короткий текст
Otel_zvezd	Числовой
Kolich_mest	Числовой
Kolich_ekskyrs	Числовой
Pitanie_raz	Числовой
Stoim_sytki	Числовой
Sobst_bereg_linija	Логический
Basein	Логический
Restoran	Логический
Vid	Поле объекта OLE

Рис. 3.9. Таблиця «Otel»

Otel

Имя поля	Тип данных
Kart	Числовой
Minibar	Логический
TV	Логический
Internet	Логический
Fitness_zal	Логический
Obmen_valut	Логический
Dajving_center	Логический
Room_service	Логический
Sayna	Логический
Biznes_zal	Логический
Spa_center	Логический
Skidka	Числовой

Рис. 3.10. Таблиця «Otel»

У таблиці «Otel» – містяться інформацію про готельні комплекси, їх характеристики та послуги.

Вигляд таблиці наведено на рисунку 3.11.

Otel

Kod_otel	Kod_gor	Name	Otel_zvezd	Kolich_mest	Kolich_eksky	Pitanie_raz	Stoim_sytki	Sobst_bereg_linija	Basein
4	16	Три долини	3	25	5	2	340	☑	☐
5	68	Шамони	4	30	4	3	450	☐	☑
6	16	Долина Гранд Русс	3	10	5	2	300	☑	☑
7	16	Валь дізер	5	28	7	1	280	☑	☐
8	16	Ле Дез Альп	3	17	2	3	200	☐	☑
9	14	Бретані	3	15	1	3	185	☑	☑
10	15	Брест	4	20	2	1	456	☐	☑
12	17	Des Alpes	5	22	3	2	235	☐	☑

Рис. 3.11. Дані про готельні комплекси, їх характеристики та послуги

Опис властивостей стовпчиків таблиці «Otel» наведено у табл. 3.5.

Таблиця 3.5

Опис властивостей стовпчиків таблиці «Otel»

Найменування	Поле	Тип поля	Довжина	Первинний ключ	Обов'язкове поле	Індексне поле
Унікальний код готелю	Kod_otel	Лічильник	Довге ціле	+	Так	ІНД
Код міста	Kod_gor	Ціле число	Довге ціле	-	Ні	-
Назва готелю	Name	Текстове	50	-	Ні	-
Кількість зірок готелю	Otel_zvezd	Байт	1	-	Ні	-
Кількість місць у готелі	Kolich_mest	Ціле число	Довге ціле	-	Ні	-
Кількість екскурсій	Kolich_ekskyrs	Ціле число	Довге ціле	-	Ні	-
Тип харчування (кількість разів)	Pitanie_raz	Ціле число	Довге ціле	-	Ні	-
Вартість проживання за добу	Stoim_sytki	Ціле число	Довге ціле	-	Ні	-
Наявність власної берегової лінії	Sobst_bereg_linija	Логічний	1	-	Ні	-
Наявність басейну	Basein	Логічний	1	-	Ні	-
Наявність ресторану	Restoran	Логічний	1	-	Ні	-
Зображення або інший OLE-об'єкт	Vid	OLE			Ні	-
Номер карти (legacy field)	Kart	Ціле число	Довге ціле	-	Ні	-
Наявність мінібару	Minibar	Логічний	1	-	Ні	-
Наявність телевізора	TV	Логічний	1	-	Ні	-
Наявність інтернету	Internet	Логічний	1	-	Ні	-
Наявність фітнес-залу	Fitness_zal	Логічний	1	-	Ні	-
Наявність обміну валюти	Obmen_valut	Логічний	1	-	Ні	-
Наявність дайвінг-центру	Dajving_center	Логічний	1	-	Ні	-
Наявність прибирання номеру	Room_service	Логічний	1	-	Ні	-

Продовження таблиці 3.5

1	2	3	4	5	6	7
Наявність сауни	Sayna	Логічний	1	-	Ні	-
Наявність бізнес-залу	Biznes_zal	Логічний	1	-	Ні	-
Наявність SPA-центру	Spa_center	Логічний	1	-	Ні	-
Розмір знижки (%)	Skidka	Числове	Довге ціле	-	Ні	-

Таблиця «Strana» складається з 2 полів, які містять в собі наступну інформацію (Рис. 3.12.):

- Унікальний ідентифікатор країни
- Назва країни

Рис. 3.12. Таблиця «Strana»

У таблиці «Strana» – містить назву країни.

Вигляд таблиці наведено на рисунку 3.13.

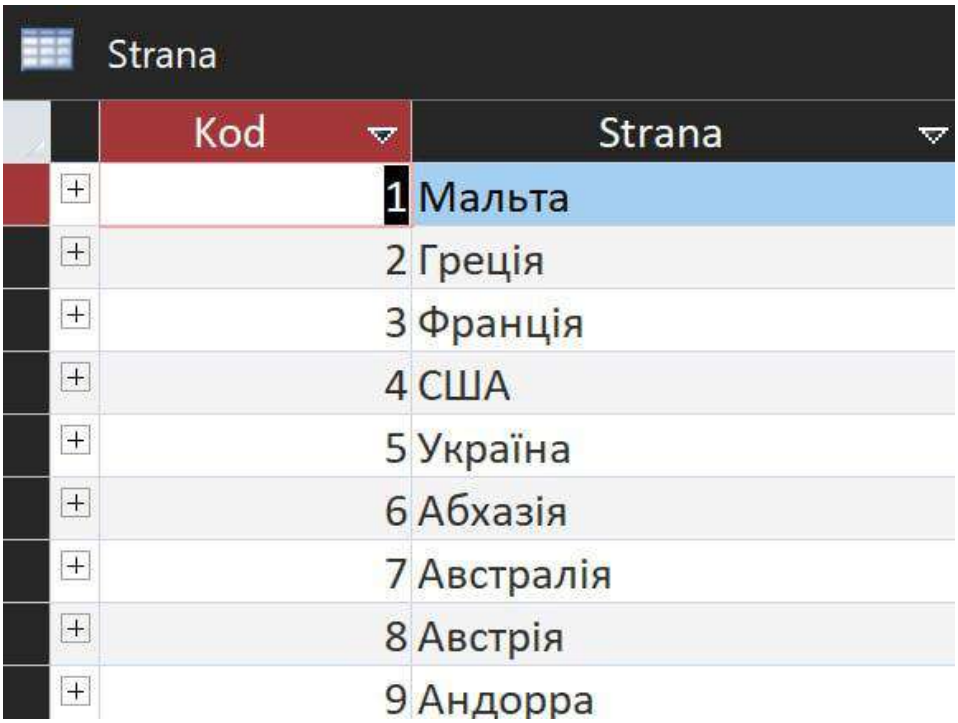


Рис. 3.13. Дані країн

Опис властивостей стовпчиків таблиці «Strana» наведено у табл. 3.6.

Таблиця 3.6

Опис властивостей стовпчиків таблиці «Strana»

Найменування	Поле	Тип поля	Довжина	Первинний ключ	Умовне значення	Обов'язкове поле	Індексне поле
Унікальний ідентифікатор країни	Kod	Лічильник	Довге ціле	+	-	Так	ІНД
Назва країни	Strana	Текстове	50	-		Ні	+

Таблиця «Tip_nomera» складається з 3 полів, які містять в собі наступну інформацію (Рис. 3.14.):

- Унікальний ідентифікатор типу номеру
- Назва типу номеру
- Розмір доплати

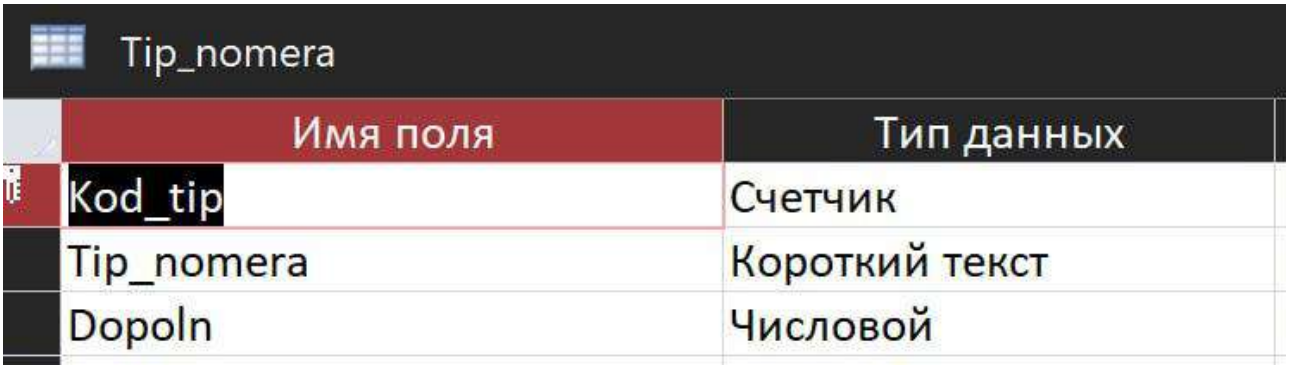
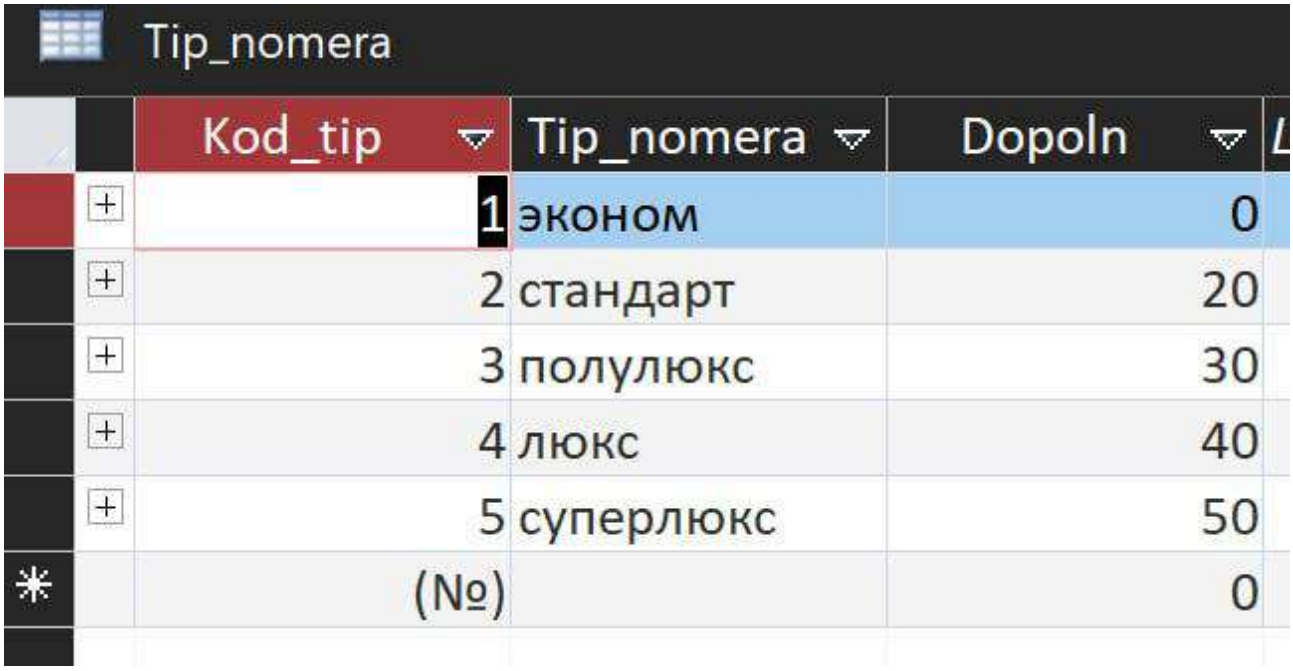


Рис. 3.14. Таблиця «Tip_nomera»

У таблиці «Tip_nomera» – містяться дані типу номеру (назва та доплату). Вигляд таблиці наведено на рисунку 3.15.


Tip_nomera

	Kod_tip	Tip_nomera	Dopoln
⊞	1	эконом	0
⊞	2	стандарт	20
⊞	3	полулюкс	30
⊞	4	люкс	40
⊞	5	суперлюкс	50
*	(№)		0

Рис. 3.15. Дані номеру (назва та доплату)

Опис властивостей стовпчиків таблиці «Tip_nomera» наведено у табл. 3.7.

Таблиця 3.7

Опис властивостей стовпчиків таблиці «Tip_nomera»

Найменування	Поле	Тип поля	Довжина	Первинний ключ	Умовне значення	Обов'язкове поле	Індексне поле
Унікальний ідентифікатор типу номеру	Kod_tip	Лічильник	Довге ціле	+	-	Так	ІНД
Назва типу номеру	Tip_nomera	Текстове	255	-		Ні	+
Розмір доплати	Dopoln	Числове	Довге ціле	-	0	Ні	-

Таблиця «Tyri» складається з 10 полів, які містять в собі наступну інформацію (Рис. 3.16.):

- Унікальний ідентифікатор туру
- Загальна вартість туру
- Розмір знижки у відсотках

- Статус виконання туру
- Валюта, в якій здійснюється оплата
- Вартість білета
- Код готелю (зовнішній ключ)
- Код міста (зовнішній ключ)
- Код типу номеру (зовнішній ключ)
- Кількість місць у номері

Tyri

Имя поля	Тип данных
Skidka	Числовой
Otel	Числовой
Valuta	Короткий текст
Stadija_vipoln	Короткий текст
Stoim_bileta	Числовой
Tip_nomera_kod	Числовой
Vid_nomera_mest	Числовой

Рис. 3.16. Таблиця «Tyri»

У таблиці «Tyri» – містяться дані про тур.

Вигляд таблиці наведено на рисунку 3.17.

Tyri

Kod_tura	Kod_gor	Stoimost	Skidka	Otel	Valuta	Stadija_vipol	Stoim_bileta	Tip_nomera_	Vid_nomera_
38	16	490	10	4	EUR	выполнено	45	2	3
40	68	500	5	5	USD	отложено	50	3	2
41	75	600	6	111	USD	выполнено	45	1	1
42	73	400	7	109	UAN	отложено	40	2	3
43	14	300	10	9	UAN	выполнено	30	1	1
44	15	350	5	10	UAN	выполнено	35	1	2
45	17	500	5	12	UAN	выполнено	30	2	2
46	72	650	5	108	USD	выполнено	40	2	3
47	18	500	5	14	UAN	выполнено	30	2	1

Рис. 3.17. Дані туру

Опис властивостей стовпчиків таблиці «Tyri» наведено у табл. 3.8.

Таблиця 3.8

Опис властивостей стовпчиків таблиці «Tyri»

Найменування	Поле	Тип поля	Довжина	Первинний ключ	Обов’язкове поле	Індексне поле
Унікальний ідентифікатор туру	Kod_tura	Лічильник	Довге ціле	+	Так	ІНД
Загальна вартість туру	Stoimost	Числове	Довге ціле	-	Так	-
Розмір знижки у відсотках	Skidka	Числове	Довге ціле	-	Ні	-
Статус виконання туру	Stadija_vipoln	Короткий текст	15	-	Ні	-
Валюта, в якій здійснюється оплата	Valuta	Короткий текст	5	-	Так	-
Вартість білета	Stoim_bileta	Числове	Ціле	-	Ні	-
Код готелю (зовнішній ключ)	Otel	Числове	Довге ціле	-	Так	+
Код міста (зовнішній ключ)	Kod_gor	Числове	Довге ціле	-	Ні	-
Код типу номеру (зовнішній ключ)	Tip_nomera_kod	Числове	Довге ціле	-	Ні	-
Кількість місць у номері	Vid_nomera_mest	Числове	Довге ціле	-	Ні	-

Висновки до розділу 3

Даний розділ містить детальний аналіз структури бази даних туристичної системи, яка є ключовим елементом для ефективного управління готельним бізнесом та туристичними пакетами.

Проведений аналіз охоплює основні сутності системи та їх взаємозв'язки, що забезпечує цілісне уявлення про інформаційну архітектуру проекту.

Розділ розпочинається з аналізу таблиці "Otel", яка є центральною сутністю системи.

Висновки підкреслюють важливість комплексного опису характеристик готелів, включаючи їх класифікацію за зірками, набір послуг та вартість проживання.

Особливу увагу приділено полю "Vid", яке містить OLE-об'єкти (ймовірно, фотографії готелів), що є важливим елементом візуалізації для клієнтів.

Важливим аспектом розділу є аналіз таблиці "Strana", яка забезпечує систематизацію країн за допомогою унікальних кодів. Висновки вказують на необхідність використання первинних ключів для забезпечення цілісності даних при зв'язку з іншими таблицями системи.

Особливе місце займає аналіз таблиці "Tip_nomera", яка реалізує систему класифікації номерів у готелях.

Висновки підкреслюють важливість цієї сутності для формування вартісної політики та розрахунків з клієнтами, зокрема через поле "Dopoln", яке визначає розмір доплати за покращений номер.

Завершальним елементом аналізу є таблиця "Tyri", яка інтегрує всі попередні сутності.

Висновки демонструють складність взаємозв'язків між таблицями через систему зовнішніх ключів, що забезпечує цілісність даних та можливість комплексного аналізу туристичних пакетів.

Загальний висновок розділу підкреслює, що правильно спроектована структура бази даних є фундаментом для ефективного функціонування туристичної системи.

Оптимально побудовані зв'язки між таблицями, чітко визначені первинні та зовнішні ключі, а також продумана система класифікації дозволяють забезпечити швидкий доступ до інформації, точність розрахунків та зручність обслуговування клієнтів.

Ця структура створює міцну основу для подальшого розвитку системи та впровадження додаткових функціональних можливостей.

РОЗДІЛ 4
РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ЗАДАЧІ

4.1. Проектування розробки програмного забезпечення

Розробка програмного забезпечення для автоматизації діяльності туристичних фірм вимагала системного підходу до проектування структури застосунку, вибору оптимальних інструментів та реалізації візуального й функціонального інтерфейсу.

У процесі проектування було реалізовано низку технічних рішень, які забезпечили надійність, зручність і гнучкість ПЗ.

1. Вибір мови програмування та середовища розробки

Основною мовою програмування обрано C++, що забезпечує високу продуктивність, контроль над пам’яттю та широкі можливості роботи з графікою, файлами та зовнішніми бібліотеками.

Як інтегроване середовище розробки (IDE) використано Microsoft Visual Studio — потужну платформу для створення десктопних застосунків, яка підтримує налагодження, управління ресурсами, візуальне проектування форм і додавання зовнішніх бібліотек (DLL, OpenGL тощо). [4]

Visual Studio також дозволяє працювати з багатомовним інтерфейсом, що було необхідно в проекті, де використовувались українська, російська та англійська мови в різних елементах (довідка, назви міст, готелів тощо).

2. Реалізація інтерфейсу користувача

Інтерфейс реалізований на базі Windows API та стандартних компонентів WinForms. Основне вікно містить меню з категоріями:

- Робота турфірми,
- Країни та міста,
- 3D,
- Налаштування,
- Гістограма в HTML,

- Допомога,
- Про програму.

Таке меню дозволяє швидко орієнтуватися у функціоналі застосунку. Додатково реалізовано діалогові вікна, наприклад, форма з інформацією про автора (фото + ПІБ), яка викликається з пункту «Про програму».

3. Візуалізація даних засобами OpenGL [4]

Однією з ключових функцій програми є відображення даних у вигляді тривимірної гістограми. Для цього було використано OpenGL — відкритий графічний API для роботи з 2D/3D графікою.

Гістограма демонструє порівняння вартості готелів у різних країнах, зображуючи кожен готель у вигляді об’ємного стовпчика з текстурованим малюнком (фото готелю) зверху.

Це дозволяє користувачу швидко візуально оцінити найвигідніші варіанти. Також реалізовано анімацію обертання сцени, що робить інтерфейс динамічним і привабливим.

4. Генерація HTML-звітів

Для збереження статистичної інформації у веб-форматі реалізовано модуль експорту даних у формат HTML.

Було застосовано вбудовані засоби мови C++ для роботи з файлами, а також додатково сформовано HTML-документ з використанням структури таблиці та діаграм.

Візуалізація даних у HTML здійснюється за допомогою бібліотеки Chart.js, яка дозволяє відображати інтерактивні гістограми з підказками при наведенні курсора.

Це забезпечує зручність перегляду результатів без потреби запускати сам застосунок — достатньо відкрити HTML-файл у браузері.

5. Система допомоги

Для створення довідки щодо використання програми застосовано формат CHM (Compiled HTML Help).

Така довідка реалізована за допомогою Microsoft HTML Help Workshop. Вона містить структуру з темами:

- Інструкція з використання,
- Готелі,
- Вартість туру,
- 3D-графіка,
- Налаштування тощо.

Це дозволяє користувачам легко орієнтуватися у функціях системи, не вдаючись до зовнішньої документації.

Вміст довідки створено у форматі HTML, що забезпечує простоту редагування та підтримку мультимедійних елементів (зображення, таблиці, гіперпосилання).

6. Обробка графіки

Для підготовки графічного контенту (іконки, фони, фото готелів) використовувались редактори GIMP та Adobe Photoshop. Зображення зменшувались, оптимізувались та стискались до формату PNG/JPEG для ефективного використання у формі та в OpenGL-сцені.

4.2. Опис програмного забезпечення «Tyrfirma»

Перше що бачить користувач при відкритті даного дипломного проекту – це головну форму.

Головна форма проекту має зручне меню, яке розташоване в верхній частині форми (Рис. 4.1.).

На рисунку зображено головне вікно прикладного програмного забезпечення, розробленого для автоматизації роботи туристичних агенцій.

Інтерфейс виконаний у сучасному стилі з використанням графічного фону, що імітує мальовничий тропічний пейзаж.

Такий підхід підсилює тематичний зв’язок із туризмом і створює приємне перше враження для користувача.

Основні елементи головного вікна:

Меню керування розташоване у верхній частині вікна і містить такі пункти:

- "Робота турфірми" – доступ до основного функціоналу, пов’язаного з обліком турів, клієнтів та замовлень;
- "Країни та міста" – база даних географічних напрямків;
- "3D" – візуалізація елементів або турів у тривимірному форматі (через OpenGL);
- "Налаштування" – виклик діалогових вікон для зміни шрифтів, кольорів, стилів таблиць та інтерфейсу;
- "Гістограма в HTML" – побудова графіків у форматі HTML (експорт із даних);
- "Допомога" – розділ довідки з інструкціями;
- "Про програму" – інформація про автора і ліцензію.

Графічне оформлення головного вікна реалізовано через фонове зображення, яке підкреслює характер застосування ПЗ — робота з туристичними турами, готелями та курортними зонами.

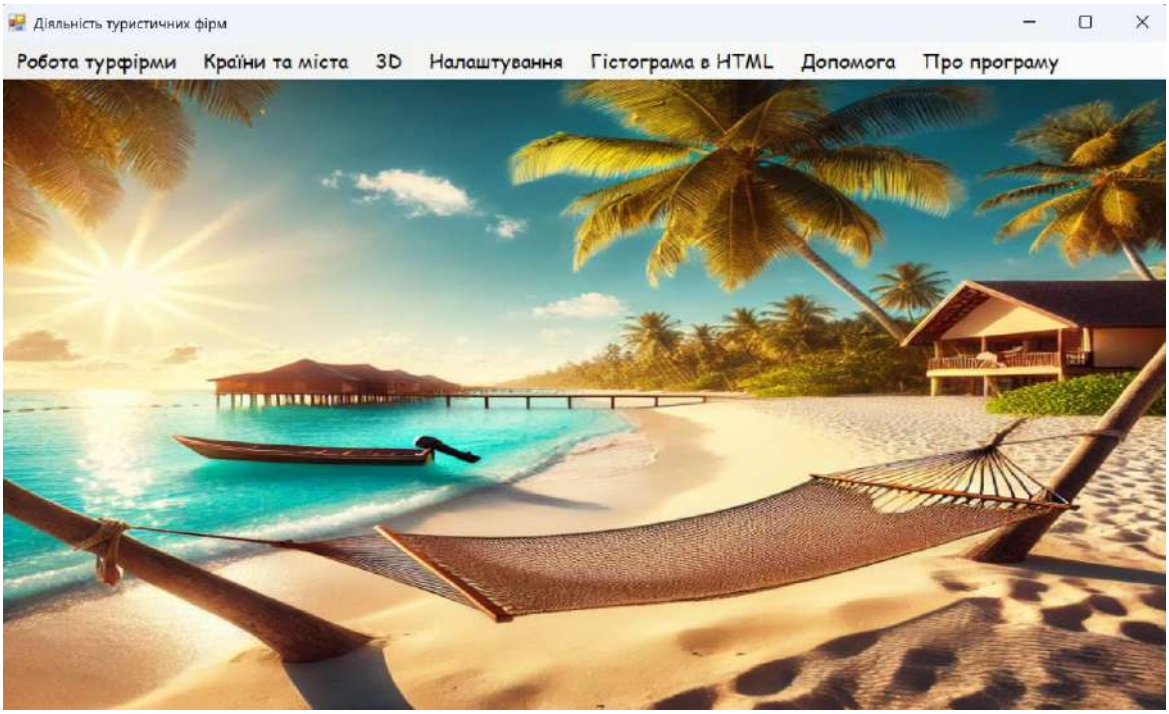


Рис. 4.1. Головна форма програми

Користування програмою відбувається за допомогою структурованого меню.

Розглянемо перший пункт головного меню «Робота турфірми». Даний пункт має випадаюче меню, яке складається з:

- Інформація по туристам – даний пункт надає змогу користувачу переглянути та внести зміни до вже існуючих записів у БД.

  Записи містять інформацію про клієнтів, тури, готелі та квитанції сплати.
- Готелі – даний пункт дозволяє користувачу обрати країну, місто та готель. Після того, як користувач визначиться він може переглянути детальну інформацію про готель, та які послуги він може надати.
- Доступні готелі – даний пункт меню дає змогу користувачу переглянути інформацію про готель на період туру.
- Вартість туру – даний пункт дозволяє користувачу переглянути вартість туру.

Вигляд випадаючого меню наведено нижче на рисунку 4.2.

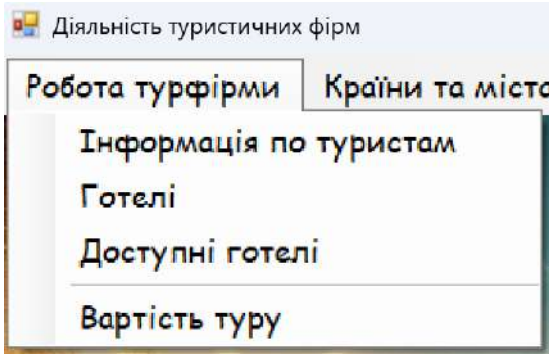


Рис. 4.2. Випадаюче меню пункту «Робота турфірми»

Тепер розберемо більш детально кожен з пунктів випадаючого меню.

1. «Інформація по туристам»

Даний пункт відкриває іншу форму програми, нижче (Рис. 4.3.) наведено її вигляд.

Інформація, яку користувач може переглянути поділяється на чотири категорії:

- Клієнти – при виборі цієї категорії в таблиці відображається наступна інформація: прізвище та ім’я, дана народження, віза та зарплатня;
- Готелі – при виборі цієї категорії в таблиці відображається наступна інформація: назва готелю, країна та місто в якому він знаходиться, його клас, кількість місць, кількість екскурсій, його вартість та додаткові послуги;
- Договори на тури – при виборі цієї категорії в таблиці відображається наступна інформація: номер договору ,дати початку та кінця туру, країна та місто, вартість розписана по критеріям, готель, тип та вид номеру, додаткові послуги, наявність знижки;
- До оплати клієнту – при виборі цієї категорії в таблиці відображається наступна інформація: прізвище, інформація про готель, вартість, знижка, вартість білету, загальна сума до сплати

Між категоріями можна переходити необмежену кількість разів. Також можна додати та видалити записи, оновити таблицю, експортувати таблицю до MS Excel та HTML.

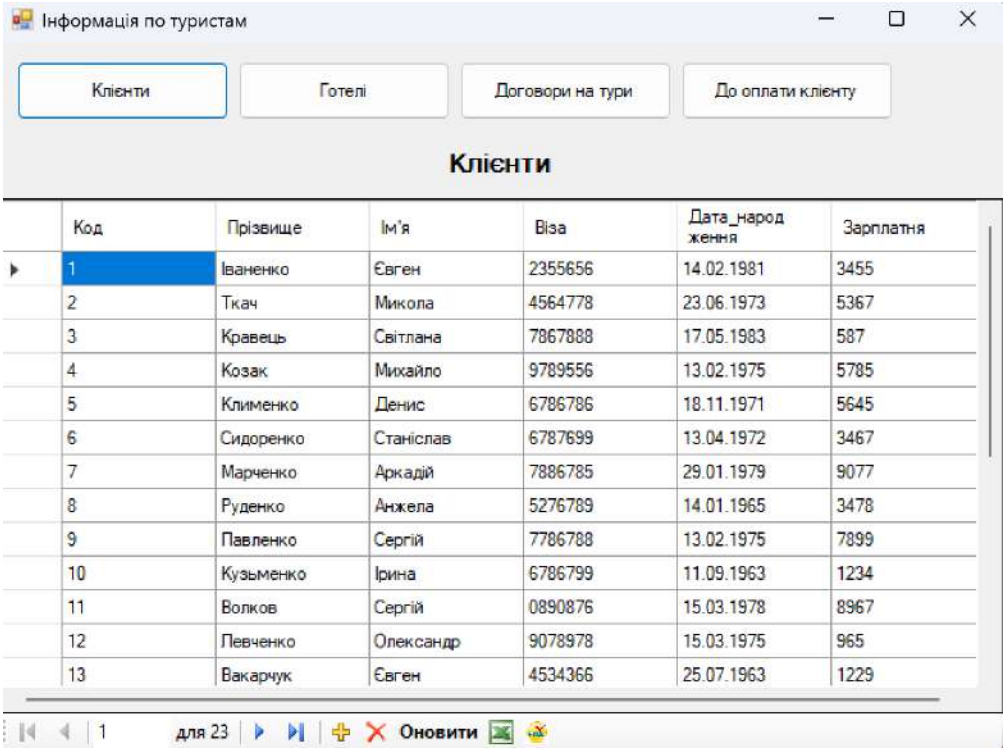


Рис. 4.3. Форма «Інформація по туристам»

2. «Готелі»

Даний пункт відкриває іншу форму, на якій користувачу слід обрати характеристики готелю (Рис. 4.4.):

- Країна;
- Місто;
- Назва готелю.

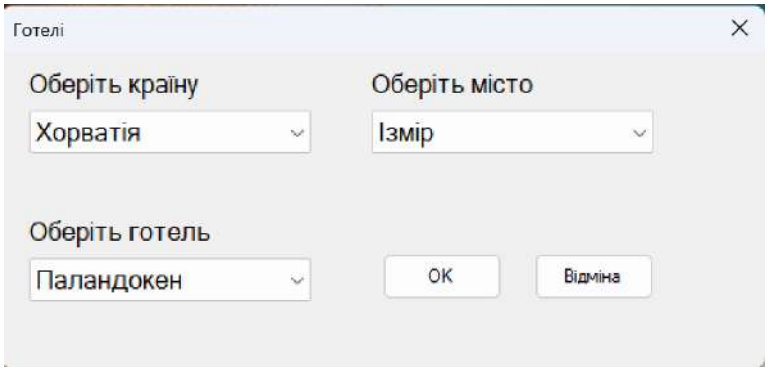


Рис. 4.4. Форма «Готелі»

Після того, як користувач визначився з готелем слід натиснути кнопку «ОК».

Відкриється інша форма, яка містить в собі інформацію про готель.

Вигляд такої інформативної форми продемонстровано на рисунку 4.5.

Рис. 4.5. Форма інформації про готель

3. «Доступні готелі»

Даний пункт меню відкриває нову форму (Рис. 4.6.), в якій користувачу слід обрати:

- Термін перебування в готелі(вказавши дату заїзду та дату виїзду)
- Обрати країну та місто
- Вказати діапазон вартості номеру(вартість вказується за добу);
- Вказати вид та тип номеру.

Кнопка «Відміна» має такий самий функціонал, як і кнопка «×»(розташована в правому верхньому кутку вікна) – вони закривають поточну форму.

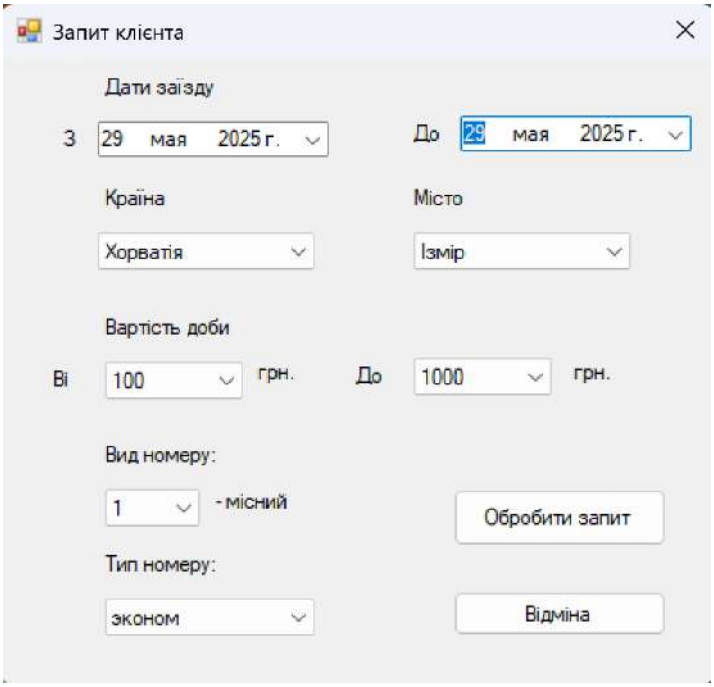


Рис. 4.6. Форма «Доступні готелі»

Після того, як користувач вказав всі необхідні дані, слід натиснути кнопку «Обробити запит».

Програма відкриє інформативну форму, на якій в табличному вигляді надає інформацію про готель та всі послуги які він має змогу надати.

Приклад вигляду інформативної сторінки представлено нижче на рисунку 4.7.

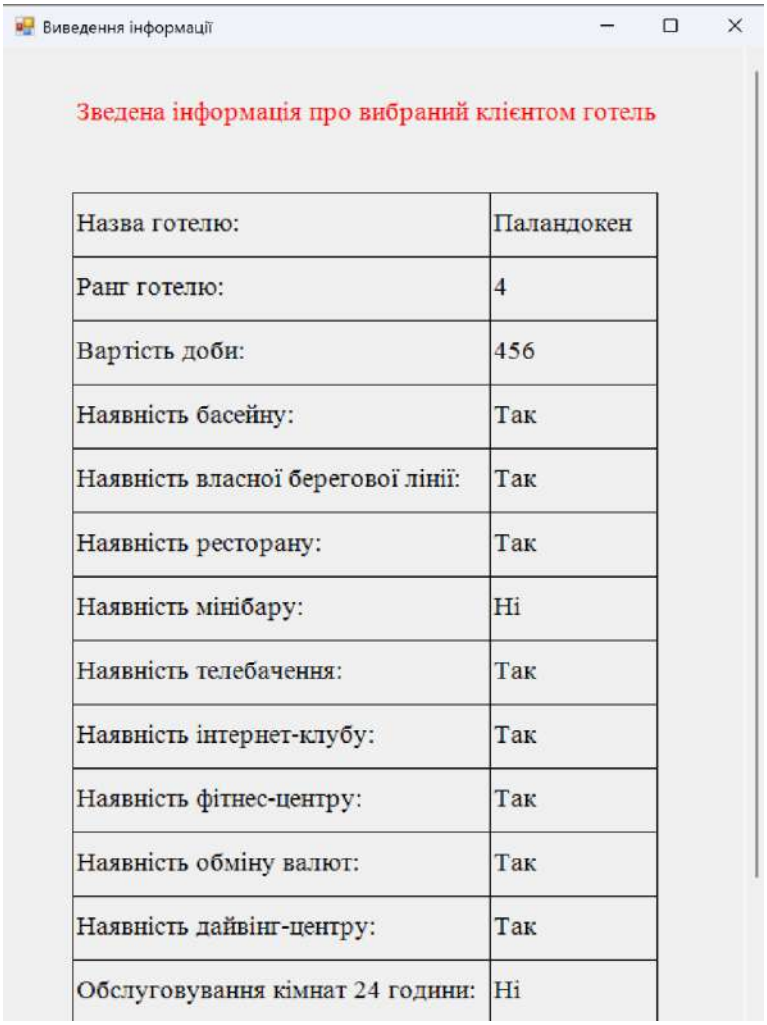


Рис. 4.7. Форма детальної інформації про готель

4. «Вартість туру»

Даний пункт меню відкриває нову форму (Рис. 4.8.), яка надає можливість користувачу розрахувати вартість відпочинку.

На цій формі користувачу слід вказати:

- Назву готелю;
- Тип номеру(економ, стандарт, напівлюкс, люкс, суперлюкс);
- Вид номеру(на 1,2 або 3 місця);
- Кількість дорослих;
- Кількість дітей;
- Кількість днів;
- Рівень обслуговування(жахливе або відмінне).

Після того як користувач вказав вище перераховані критерії, поставивши маркер біля опції «Оновити розрахунок вартості», в нижній частині форми з лівого боку можна бачити як змінюється вартість при зміні критеріїв.

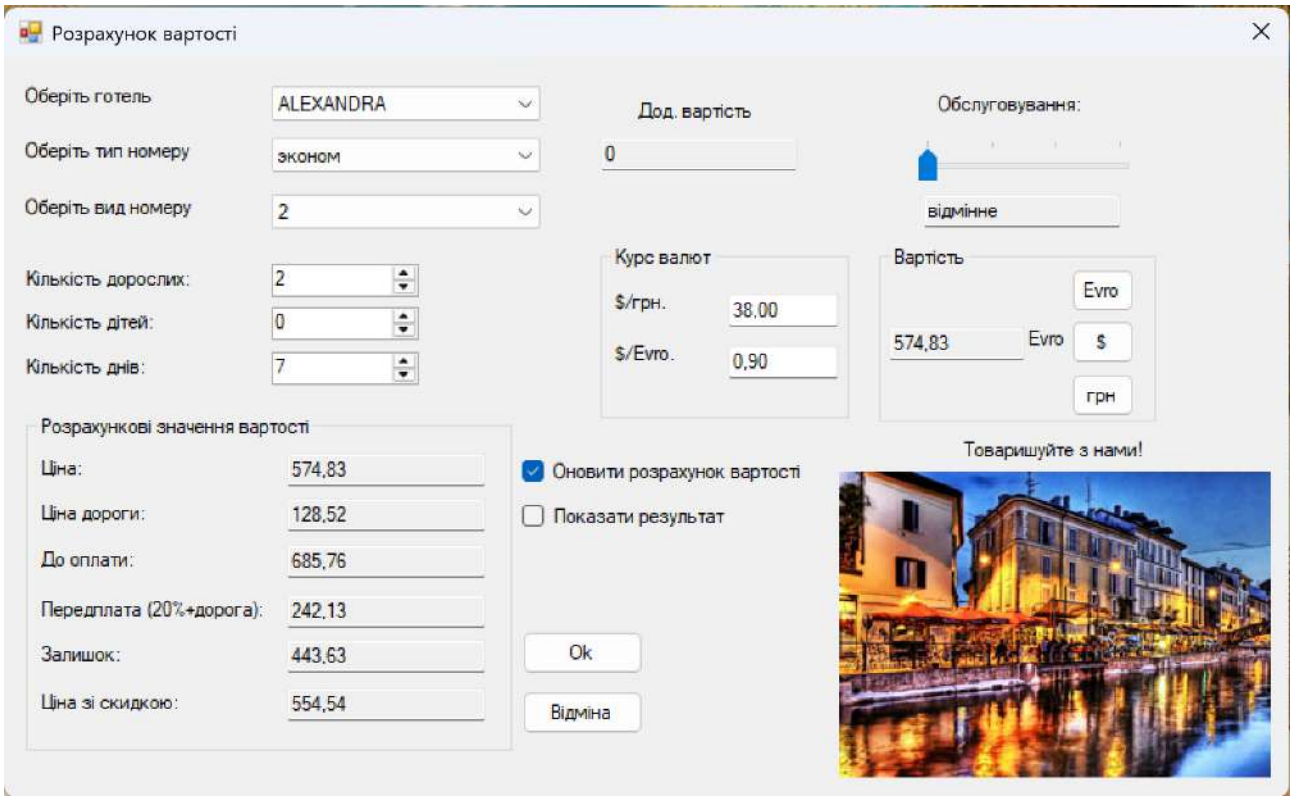


Рис. 4.8. Головне меню програми

Поставивши маркер біля опції «Показати результат» і натиснувши кнопку «ОК» можна переглянути «чек» за тур.

Коли користувач натискає на кнопку «ОК» відкривається інформативна форма (Рис. 4.9.), на якій в табличному вигляді виводиться вартість за наступними критеріями:

- Назва готелю;
- Знижка;
- Ціна зі знижкою – сума з урахуванням знижки;
- Вартість туру – сума за перебування в готелі;
- Вартість дороги – сума за квитки;
- До сплати – загальна вартість туру;
- Передплата – сума, яку клієнт вже сплатив;
- Залишок – сума, яку залишилось сплатити.

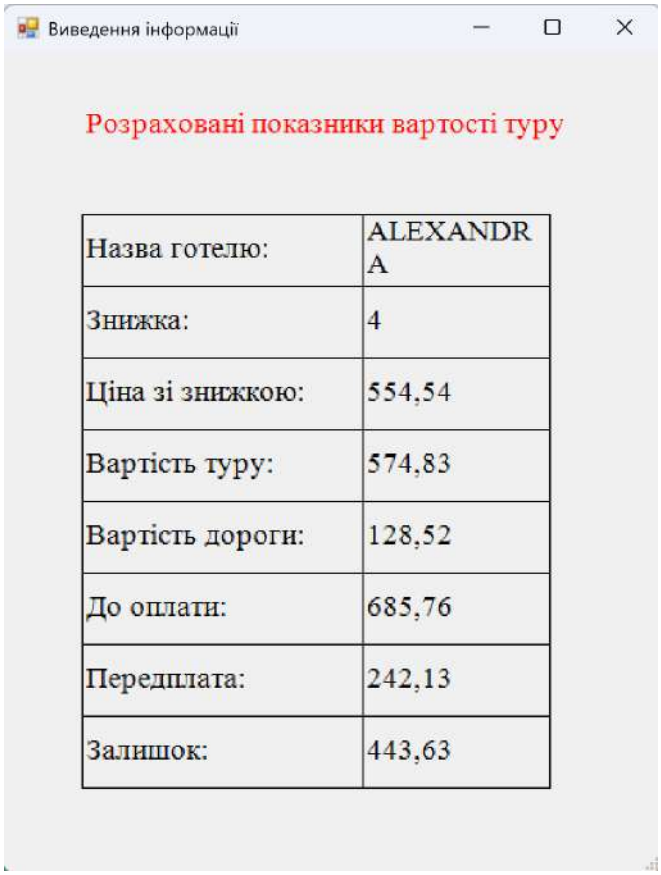


Рис. 4.9. Форма «Чек»

Наступний пункт головного меню «Країни та міста», він також має випадаюче меню, яке складається з двох пунктів:

1. Країни.
2. Міста.

Вигляд випадаючого меню пункту «Країни та міста» наведено нижче (Рис. 4.10.).

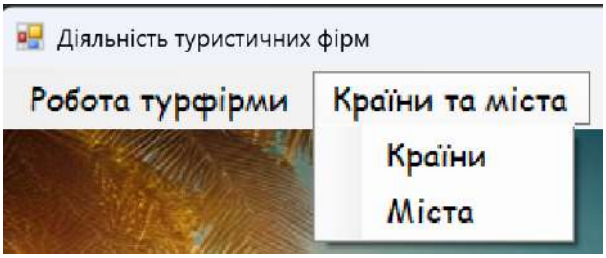


Рис. 4.10. Випадаюче меню «Країни та міста»

1. «Країни»

Даний пункт меню відкриває нову форму (Рис. 4.11.). На цій формі користувач має змогу:

- Переглянути список всіх країн;

- Додати нову країну до списку;
- Змінити назву вже існуючої країни в списку;
- Зберегти або видалити зміни.

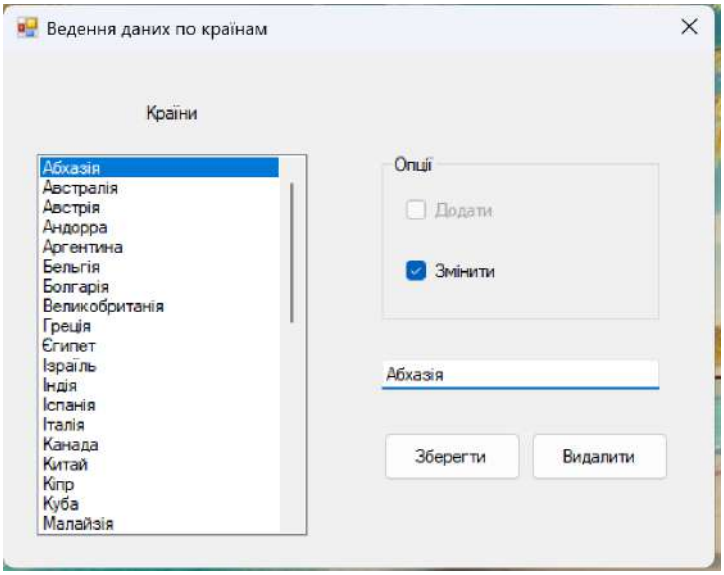


Рис. 4.11. Форма «Країни»

2. «Міста»

Даний пункт меню відкриває нову форму (Рис. 4.12.). На цій формі користувач має змогу:

- Переглянути список всіх країн;
- Додати нове місто до списку (обравши країну, в якій воно знаходиться);
- Змінити назву вже існуючого міста в списку;
- Зберегти або видалити зміни.

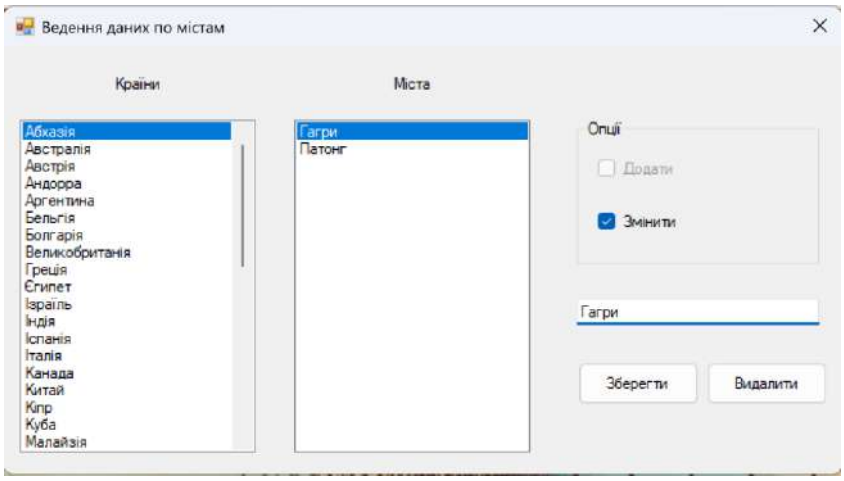


Рис. 4.12. Форма «Міста»

Наступний пункт головного меню «3D». Даний пункт меню відкриває нове вікно програми.

В цьому вікні відображено гістограму з використанням 3D-графіки (Рис. 4.13.).

Нижче на рисунку представлено приклад побудови об’ємної гістограми з використанням графічної бібліотеки OpenGL, яка візуалізує вартість проживання в готелях за добу. Візуалізація створена як частина розробленого програмного забезпечення для організації туристичних турів.

Особливості реалізації:

- Кожен стовпчик відображає вартість конкретного готелю у гривнях;
- Над стовпчиками розміщено фотографії відповідних готелів, що полегшує асоціативне сприйняття даних;
- Для кожного стовпчика виводиться точне значення ціни, наприклад, "456.00";
- Стовпчики мають 3D-ефект з градієнтними текстурами, що створює об’ємну візуалізацію;
- Тло гістограми витримано у темно-сірому кольорі з контрастною «земною» основою, що візуально відділяє графік від фону;
- Колір стовпчиків варіюється, підкреслюючи індивідуальність кожного готелю.

Цей підхід демонструє можливості графічного рендерингу з текстуруванням та освітленням у середовищі OpenGL, забезпечуючи інтерактивність та естетичну привабливість при поданні статистичних даних.

Рис. 4.13. Головне вікно «3D»

Наступний пункт головного меню «Налаштування», має випадаюче меню, яке зображено на рисунку 4.14.

У розробленому програмному забезпеченні для організації туристичних турів передбачено можливість гнучкого налаштування зовнішнього вигляду елементів таблиці, яка містить інформацію про готелі, ціни, тривалість туру та інші дані.

Зокрема, реалізовано програмний механізм для зміни шрифтів та кольорів заголовків таблиці, стовпців і загального тексту таблиці через інтерфейс користувача.

Використання стандартних діалогових вікон

У додатку на мові C# реалізовано виклик стандартних діалогових вікон Windows Forms:

- FontDialog – для вибору шрифтів (назва, розмір, стиль);
- ColorDialog – для вибору кольорів тексту або фону.

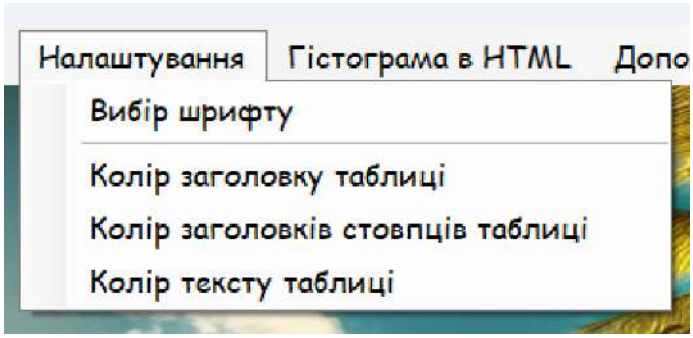


Рис. 4.14. Випадаюче меню «Налаштування»

На рисунку 4.15. зображено стандартне діалогове вікно вибору шрифту в операційній системі Windows, яке використовується в розробленому програмному забезпеченні для налаштування зовнішнього вигляду тексту в таблицях.

Це вікно викликається програмно за допомогою компонента FontDialog у середовищі C# Windows Forms. [1]

Діалог дозволяє користувачеві:

- вибрати назву шрифту (ліворуч);

- встановити стиль написання (звичайний, курсив, напівжирний тощо);
- задати розмір шрифту;
- застосувати додаткові ефекти (закреслений, підкреслений);
- переглянути приклад оформлення в полі «Зразок»;
- обрати набір символів (наприклад, західноєвропейський).

Після підтвердження натисканням кнопки «ОК», вибрані параметри застосовуються до тексту в таблицях — як до заголовків стовпців, так і до основного тексту.

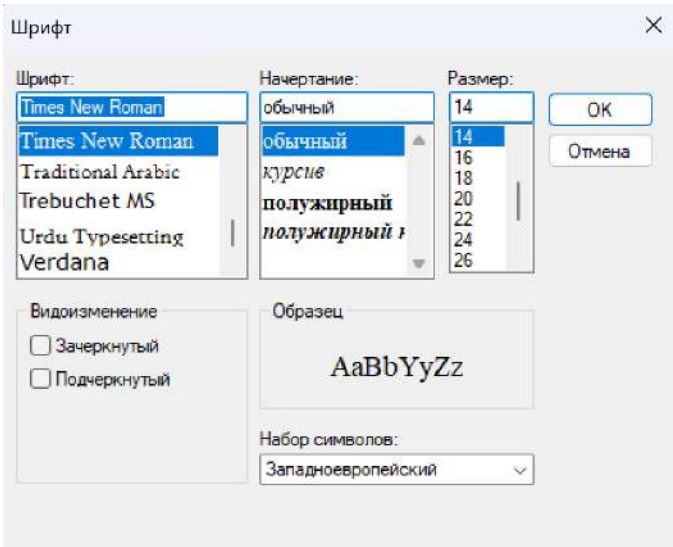


Рис. 4.15. Діалогове вікно налаштувань шрифту

Нижче представлено діалогове вікно вибору кольору (стандартний режим).

На рисунку 4.16. представлено базову версію стандартного діалогового вікна вибору кольору у Windows, яке викликається через компонент ColorDialog у середовищі C#. Це вікно використовується в додатку для вибору кольору тексту або фону таблиць.

Інтерфейс містить:

- набір основних кольорів, доступних для швидкого вибору;
- порожній блок додаткових кольорів, який можна заповнити користувацькими відтінками;
- кнопки «ОК» та «Отмена» для підтвердження або скасування вибору;

- кнопку «Определить цвет >>», яка відкриває розширене меню кольорової палітри.

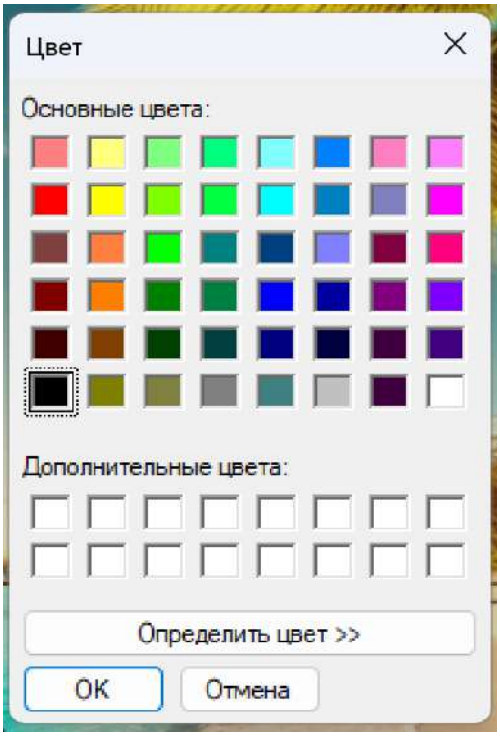


Рис. 4.16. Діалогове вікно налаштувань кольору

На рисунку 4.17., який наведено нижче зображено розгорнуту версію діалогового вікна, яка відкривається після натискання кнопки «Определить цвет».

Це дозволяє користувачеві точно налаштувати бажаний колір за допомогою кольорової палітри, а також ввести числові значення компонентів кольору (RGB, яскравість, контраст, відтінок).

Користувач має можливість:

- визначити власний відтінок кольору, вибираючи зі спектра;
- ввести значення вручну для досягнення точності;
- додати колір до набору додаткових кольорів для повторного використання;
- бачити попередній перегляд обраного кольору.

Обидва діалогові вікна активно застосовуються в інтерфейсі програми — наприклад, для налаштування кольору заголовків таблиць, тексту в комірках або фону.

Це покращує користувацький досвід, дозволяючи персоналізувати вигляд виведеної інформації.

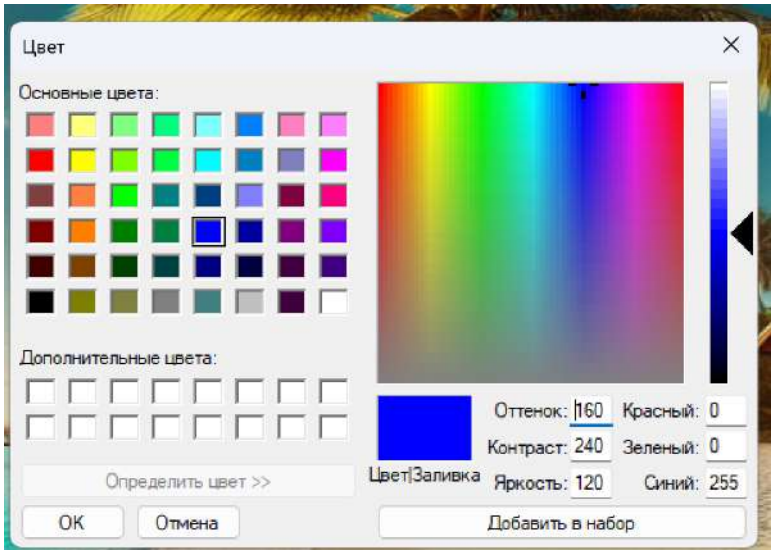


Рис. 4.17. Розширене діалогове вікно налаштувань кольору

Наступний пункт меню – це «Гістограма в HTML».

На рисунку 4.18. зображено приклад візуалізації даних у форматі HTML, яка генерується програмним забезпеченням в результаті експорту таблиці з цінами готелів.

Виведена стовпчаста діаграма відображає вартість проживання в різних готелях за одну добу (у гривнях).

Особливості реалізації:

- Кожен стовпчик відповідає окремому готелю, назва якого виводиться по осі X;
- По осі Y відображається вартість (у грн);
- При наведенні курсору на стовпчик відображається підказка (tooltip) з назвою готелю та точною вартістю;
- Легенда у верхній частині діаграми пояснює, що саме представляє візуалізоване значення ("Ціна готелів за добу");
- Дизайн графіка підтримує темну тему оформлення, що відповідає загальному стилю інтерфейсу програми;

- Візуалізація реалізована з використанням сучасних HTML-технологій (наприклад, JavaScript-бібліотеки [Chart.js, Plotly.js або аналогів — залежно від конкретної реалізації]). [6]

Ця функціональність дозволяє користувачам експортувати дані не лише у вигляді таблиці, а й у форматі зручної для сприйняття графіки, яку можна легко вбудувати в веб-сторінку або надрукувати.

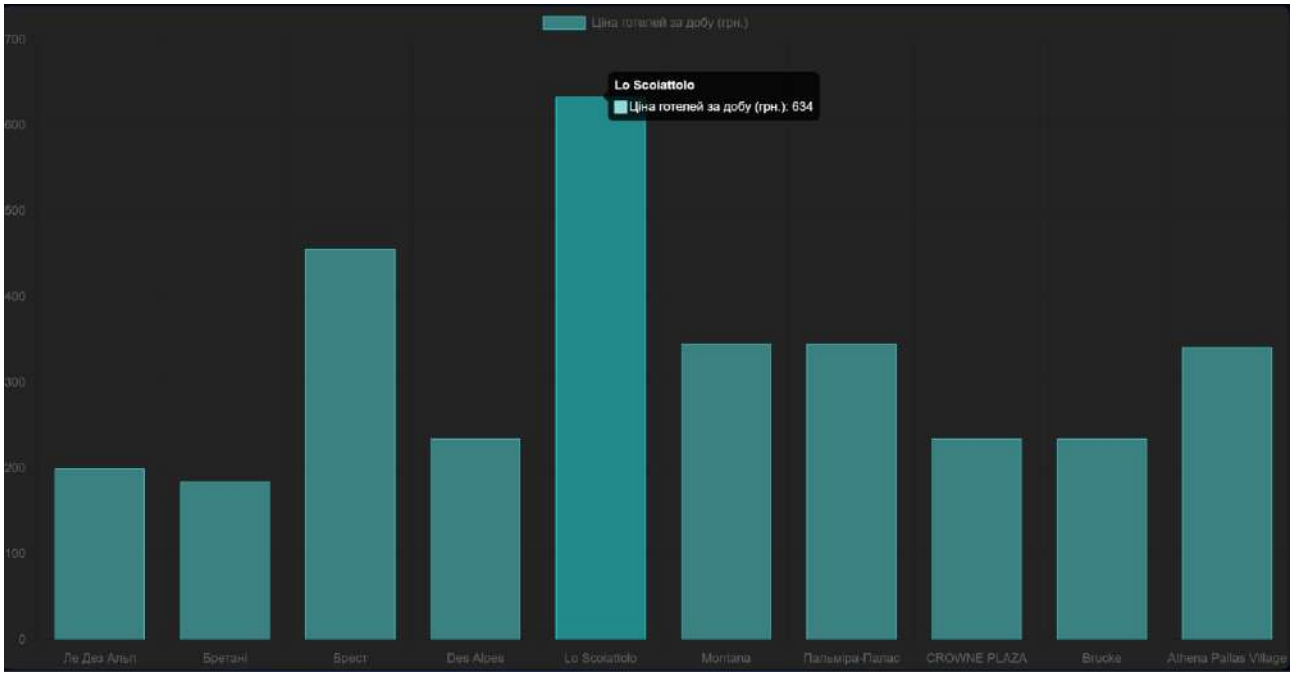


Рис 4.18. Вигляд експорту в HTML

Наступний пункт головного меню «Допомога», коли користувач клікає на нього відкривається нове вікно (Рис. 4.19.).

Це вікно є довідковою системою програмного забезпечення, реалізованою за допомогою стандартного механізму HTML Help (формат .chm або подібний).

Має наступні елементи інтерфейсу:

- Заголовок вікна
- Панель інструментів: містить кнопки:
  - Скрыть / Показать зміст,
  - Назад (перехід на попередню сторінку),
  - Печать (друк вибраної сторінки),
  - Параметры (налаштування вигляду чи функціоналу довідки).

- Зміст (ліва панель): структуроване дерево розділів довідки українською та російською мовами:

Має одну гілку «Інструкція з використання програми», яка в свою чергу містить в собі:

- Інформація по туристам
- Готелі
- Доступні готелі
- Вартість тура
- Країни та міста
- 3D-графіка
- Налаштування

Цей вміст дозволяє користувачу швидко орієнтуватися в ключових функціях програми та отримувати довідкову інформацію.

Структура довідки логічна, охоплює основні модулі програми.

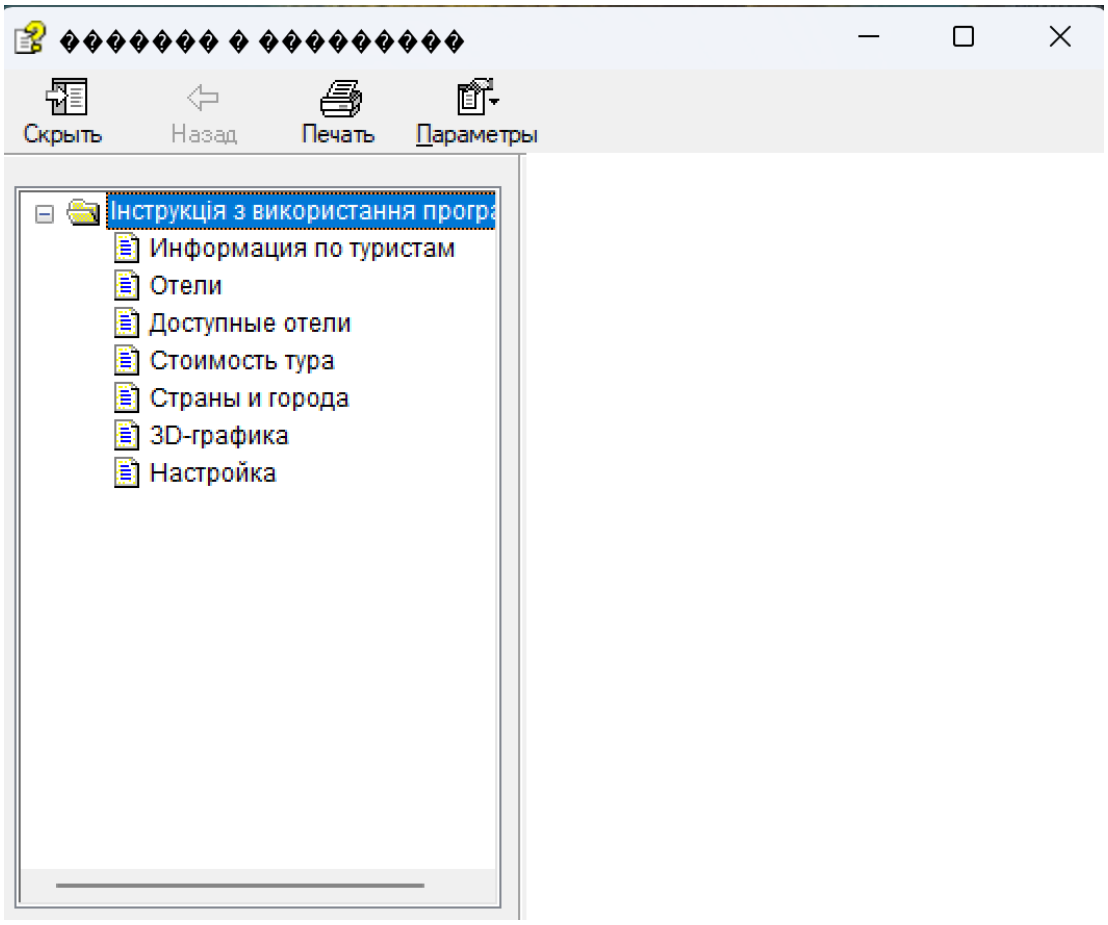


Рис. 4.19. Головне меню програми

Останній пункт головного меню «Про програму» відображає інформацію про автора програмного забезпечення.

Вигляд форми, яку запускає даний пункт меню зображено на рисунку 4.20.

Вікно про програму, відкривається за відповідним пунктом меню і призначене для ознайомлення користувача з авторськими правами.

Форма містить:

- Зображення готелю, яке асоціюється з тематикою програмного забезпечення (організація туристичних турів);
- Текстовий напис:

  «Авторські права: Дегтярьов Артем» — виконаний чітким шрифтом з виділенням (наприклад, напівжирним), що акцентує увагу користувача;
- Кнопка "Ok", що закриває діалогове вікно після ознайомлення з інформацією.

Інтерфейс реалізований у мінімалістичному стилі, що відповідає загальному дизайну програми: відсутність зайвих елементів, логічна структура і доброзичливе інформативне оформлення.

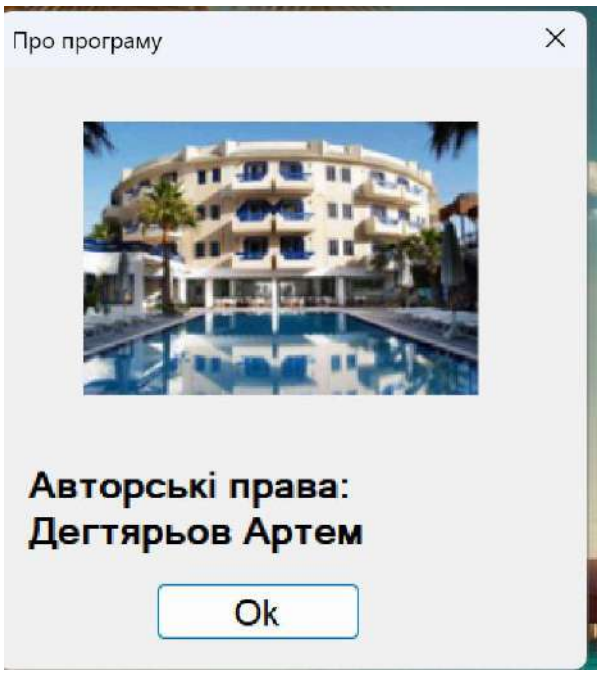


Рис 4.20. Вікно авторського права

Таким чином, у даному пункті було детально охарактеризовано реалізоване програмне забезпечення «Tyrfirma», його інтерфейсні компоненти, функціональні можливості та логіку взаємодії з користувачем.

Продемонстровано основні елементи інтерфейсу, такі як головне вікно, меню керування, діалогові вікна налаштувань, механізми побудови графіків і генерації звітів.

Особливу увагу приділено гнучкості налаштувань таблиць, підтримці форматування та інтеграції з графічною бібліотекою OpenGL. Описані компоненти забезпечують повноцінне виконання завдань, поставлених на етапі проектування системи.

Висновки до розділу 4

У даному розділі було розглянуто графічний інтерфейс користувача програмного забезпечення діяльності туристичних фірм.

Зокрема, проаналізовано структуру головного вікна, елементи меню, а також реалізацію діалогових вікон для налаштування шрифтів, кольорів і побудови гістограм.

Програма має інтуїтивно зрозумілий інтерфейс, що забезпечує швидкий доступ до ключових функцій: ведення бази турів, міст та країн, побудова графіків у HTML та OpenGL.

Візуальне оформлення головного вікна гармонійно поєднується з тематикою туризму, що сприяє зручності та естетичному сприйняттю системи.

Розроблений інтерфейс відповідає основним вимогам до сучасного прикладного програмного забезпечення — забезпечує простоту використання, функціональну повноту та візуальну привабливість.

Це створює сприятливі умови для подальшого використання програми у роботі туристичних компаній.

ВИСНОВКИ

В межах кваліфікаційної роботи було створено програмне забезпечення, призначене для автоматизації процесів в туристичній галузі.

Реалізована система дозволяє працювати з інформацією про клієнтів, готелі, тури та супровідні послуги, що забезпечує швидкий доступ до актуальних даних і зручність під час роботи з великою кількістю записів.

Результатом виконання поставлених завдань стало створення функціонального десктопного додатку з графічним інтерфейсом, що охоплює такі можливості: введення та редагування інформації, побудова інтерактивної статистики, налаштування зовнішнього вигляду таблиць, а також формування звітів у форматі HTML.

Основна частина програмного забезпечення реалізована з використанням мови C# та технології Windows Forms, візуалізація графіків — з використанням OpenGL.

Особливу увагу під час розробки було приділено адаптивності інтерфейсу та зручності користувача: реалізовано можливість змінювати шрифти, кольори заголовків та тексту, керувати відображенням даних, що значно покращує візуальне сприйняття інформації.

Також реалізовано механізм створення звітів у вигляді HTML-документів із підтримкою стилізації, що робить їх придатними як для друку, так і для публікації в електронному вигляді.

Алгоритмічна складова системи розроблена з урахуванням надійності, ефективності та простоти масштабування.

Передбачено обробку помилок введення, логічні перевірки даних, оптимізовану побудову графіків і таблиць.

Система дозволяє динамічно реагувати на зміну користувацьких параметрів і зберігає стабільність роботи при активному використанні.

Таким чином, поставлену мету дипломного проєкту — створення програмного інструменту для підтримки роботи туристичної компанії — було досягнуто в повному обсязі.

Розроблений додаток виконує основні функції, необхідні для обробки туристичної інформації, її візуального представлення та створення звітів.

Отриманий результат є прикладом прикладного програмного рішення, яке може бути використане як у навчальних цілях, так і для подальшого доопрацювання та адаптації під реальні умови.

У перспективі проєкт має потенціал до вдосконалення — передбачено можливість підключення до реляційних баз даних, реалізації механізмів бронювання, розширення статистичних функцій та побудови графіків на основі збережених запитів користувачів.

Такі розширення дозволять значно розширити функціонал системи без зміни її основної архітектури.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Молчанов К. О. Основи програмування мовою C# / К. О. Молчанов. — К.: Наука і освіта, 2020. — 352 с.
2. Шилдт Г. C# 9 і .NET 5: повний посібник / Г. Шилдт. — К. : Діалектика, 2021. — 864 с.
3. Petzold C. Programming Windows Forms with C# / Charles Petzold. — Microsoft Press, 2018. — 560 p.
4. Каюмов Р. А. Програмування графіки з OpenGL / Р. А. Каюмов. — М. : БІНОМ, 2017. — 256 с.
5. Shreiner D. OpenGL Programming Guide: The Official Guide to Learning OpenGL / D. Shreiner. — Addison-Wesley, 2020. — 976 p.
6. Фленаган Д. JavaScript. Подробное руководство / Д. Фленаган. — СПб. : Питер, 2021. — 1088 с.
7. W3C. HTML5 Specification [Електронний ресурс] – Режим доступу: https://html.spec.whatwg.org
8. MSDN Documentation. Windows Forms [Електронний ресурс]. – Режим доступу: https://docs.microsoft.com/en-us/dotnet/desktop/winforms
9. Бейликов С. В. Системний аналіз та проектування інформаційних систем / С. В. Бейликов. — К. : Видавничий дім «Слово», 2019. — 400 с.
10. Глинский Я. М. Основы программной инженерии / Я. М. Глинский. — Минск : Новое знание, 2020. — 288 с.
11. ISO/IEC 14882:2014. Information technology — Programming languages — C++ [Електронний ресурс].
12. Писаренко О. В. Архітектура програмного забезпечення: навчальний посібник. — К. : КНУ, 2022. — 246 с.
13. Кузьміна С. В. Проєктування програмних систем. — Кривий Ріг: ДУЕТ, 2023. — 198 с.

ДОДАТКИ

Додаток А

Лістинг Form1

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.OleDb;
using System.Drawing;
using System.Diagnostics;
using System.Text;
using System.IO;
using System.Windows.Forms;
using Excel = Microsoft.Office.Interop.Excel;

namespace Tyrfirma
{
    public partial class Form1 : Form
    {
        OleDbConnection conn;
        //Статичні налаштування
        public static Color cvet_zag;
        public static Color cvet_hapka;
        public static Color cvet_text;
        public static Font font_text;
        stoim st;
        int kod_zap = 0;

        public Form1()
        {
            InitializeComponent();
            st = new stoim();
           //Формування об'єкту Connection
            try
            {
             string source = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=турист.mdb";
                conn = new OleDbConnection(source);
                conn.Open();
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
                return;
            }
            // Зчитування налаштувань
            try
            {
    BinaryReader bb = new BinaryReader(new FileStream("settings.ini", FileMode.Open));
                cvet_zag = Color.FromArgb(bb.ReadByte(), bb.ReadByte(), bb.ReadByte());
        cvet_hapka = Color.FromArgb(bb.ReadByte(), bb.ReadByte(), bb.ReadByte());
             cvet_text = Color.FromArgb(bb.ReadByte(), bb.ReadByte(), bb.ReadByte());
    font_text = new Font(bb.ReadString(), bb.ReadSingle(), (FontStyle)bb.ReadInt32());
                bb.Close();
                return;
            }
            catch(Exception)
            {
 MessageBox.Show("Файл settings не знайдено!\nБудуть прийняті значення за умовчанням");
            }
            cvet_zag = Color.Red;
            cvet_hapka = Color.Blue;
```

```
            cvet_text = Color.Black;
            font_text = new Font("Times New Roman", 14, FontStyle.Regular);
            kod_zap = 1;
        }
        // Універсальна функція експорту в Excel довільних даних
        public static bool ExportXLS(DataView dt)
        {
            try
            {
                Excel.Application ExcelApp = new Excel.Application();
                Excel.Workbook book = ExcelApp.Workbooks.Add(Type.Missing);
                ExcelApp.SheetsInNewWorkbook = 1;
                Excel.Worksheet sheet = (Excel.Worksheet)ExcelApp.Sheets.get_Item(1);
                sheet.Name = "Мой лист";
                Excel.Range range, range1;
                range = sheet.Cells;
                string str = (char)('A' + dt.Table.Columns.Count - 1) + "";
                //A1 - D1
      range1 = range.get_Range("A1", ((char)('A' + dt.Table.Columns.Count - 1)) + "1");
                range1.HorizontalAlignment = Excel.XlHAlign.xlHAlignCenter;
                range1.Font.Bold = true;
                int row = 1, col;
                string v;
                Excel.Border border;
                border = range1.Borders.get_Item(Excel.XlBordersIndex.xlEdgeTop);
                // Рисование шапки
                for (col = 0; col < dt.Table.Columns.Count; col++)
                {
                    v = dt.Table.Columns[col].ColumnName;
                    if (v == "") continue;
                    range.set_Item(row, col + 1, v);
                }
                row++;
                foreach (DataRowView rr in dt)
                {
                    for (col = 0; col < dt.Table.Columns.Count; col++)
                    {
                        v = rr[col].ToString();
                        if (v == "") continue;
                        double dd;
                       if (dt.Table.Columns[col].DataType.ToString() == "System.DateTime")
                            v = Convert.ToDateTime(rr[col]).ToShortDateString();
                        if (double.TryParse(v, out dd) == true)
                            range.set_Item(row, col + 1, dd);
                        else
                            range.set_Item(row, col + 1, v);
                    }
                    row++;
                }
                range1 = range.get_Range("A1", ((char)('A' + dt.Table.Columns.Count -
1)) + (dt.Count + 1).ToString());
                range1.VerticalAlignment = Excel.XlVAlign.xlVAlignCenter;
              range1.Borders.get_Item(Excel.XlBordersIndex.xlEdgeBottom).LineStyle = 1;
                range1.Borders.get_Item(Excel.XlBordersIndex.xlEdgeLeft).LineStyle = 1;
              range1.Borders.get_Item(Excel.XlBordersIndex.xlEdgeRight).LineStyle = 1;
                range1.Borders.get_Item(Excel.XlBordersIndex.xlEdgeTop).LineStyle = 1;
          range1.Borders.get_Item(Excel.XlBordersIndex.xlInsideHorizontal).LineStyle = 1;
          range1.Borders.get_Item(Excel.XlBordersIndex.xlInsideVertical).LineStyle = 1;
                range1.Font.Name = Form1.font_text.Name;
                range1.Font.Size = Form1.font_text.SizeInPoints;
```

```
                range1.Font.Color = Form1.cvet_text;
      range1 = range.get_Range("A1", ((char)('A' + dt.Table.Columns.Count - 1)) + "1");
                range1.Font.Color = Form1.cvet_hapka;
                sheet.Columns.AutoFit();
                ExcelApp.Visible = true;
            }
            catch (Exception)
            {
                return false;
            }
            return true;
        }
        //Універсальна функція скрипта для побудови діаграм в Web
        public static void SaveHtmlFile(string fileName, string title, string
chartLabel, string[] countries, double[] population)
        {
            // Формуємо HTML-контент
            string htmlContent = $@"
<!DOCTYPE html>
<html lang='uk'>
<head>
    <meta charset='UTF-8'>
    <meta name='viewport' content='width=device-width, initial-scale=1.0'>
    <title>{title}</title>
    <script src='https://cdn.jsdelivr.net/npm/chart.js'></script>
    <style>
        body {{
            font-family: Arial, sans-serif;
            text-align: center;
            margin: 0;
            display: flex;
            justify-content: center;
            align-items: center;
            height: 100vh;
            background-color: #f4f4f4;
        }}
        canvas {{
            width: 100vw !important;
            height: 100vh !important;
        }}
    </style>
</head>
<body>
    <canvas id='populationChart'></canvas>
    <script>
        const ctx = document.getElementById('populationChart').getContext('2d');
        new Chart(ctx, {{
            type: 'bar',
            data: {{
                labels: {GenerateArrayForJS(countries)},
                datasets: [{{
                    label: '{chartLabel}',
                    data: {GenerateArrayForJS(population)},
                    backgroundColor: 'rgba(75, 192, 192, 0.6)',
                    borderColor: 'rgba(75, 192, 192, 1)',
                    borderWidth: 1
                }}]
            }},
            options: {{
                responsive: true,
```

```
                maintainAspectRatio: false,
                scales: {{
                    y: {{
                        beginAtZero: true
                    }}
                }}
            }}
        }});
    </script>
</body>
</html>";
            // Записуємо HTML в файл
            File.WriteAllText(fileName, htmlContent);
        }
        private static string GenerateArrayForJS(string[] array)
        {
            return $"['{string.Join("', '", array)}']";
        }
        private static string GenerateArrayForJS(double[] array)
        {
            return $"[{string.Join(", ", array)}]";
        }
        // Універсальна функція експорту в HTML довільних даних
        // ф-я експорту будь-якої таблиці в файл html.
        // Вхід:
        // filename - им'я html-файлу в який відбувається експорт
        // dt - представлення записів DataView
        // zaglav - заголовок таблиці
        // ff - шрифт
        // cvetz - колір заголовку таблиці
        // cveth - колір заголовків стовпців таблиці
        // cvett - колір тексту комірок таблиці
        public static bool Export_Html(string filename, DataView dt, string zaglav,
Font ff, Color cvetz, Color cveth, Color cvett)
        {
            int i, kol_p;
            string vt;
            try
            {
            StreamWriter writer = new StreamWriter(filename, false, Encoding.Default);
                writer.Write("<html>\n");
                writer.Write("<head>\n");
                writer.Write("<meta http-equiv=Content-Type \"content=text/html;
charset=windows-1251\">\n");
                writer.Write("<title>");
                writer.Write("Експорт таблиці");
                writer.Write("</title>\n");
                writer.Write("<style>\n");
                writer.Write("table{border-style:solid; ");
                writer.Write("border-top-style:none;font-family:{0};font-size:{1}pt;",
ff.Name, (int)ff.SizeInPoints);
                //"{0:x2}{1:x2}{2:x2}"
                writer.Write("background-color:white;}\np.zag{color:");
                writer.Write("#{0:x2}{1:x2}{2:x2}; text-align:center; font-weight:bold;
font-family:{3}; font-size:{4}pt; color:#{0:x2}{1:x2}{2:x2};",
                    cvetz.R, cvetz.G, cvetz.B, ff.Name, (int)ff.SizeInPoints);
                writer.Write("}\n");
                writer.Write("td{border-style:none;border-right-style:solid;border-top-
style:solid;border: solid windowtext 1px;}\n");
                writer.Write("td.text{color:");
```

```
                writer.Write("#{0:x2}{1:x2}{2:x2};", cvett.R, cvett.G, cvett.B);
                writer.Write("}\n");
                writer.Write("td.shapka{color:");
    writer.Write("#{0:x2}{1:x2}{2:x2}; font-weight:bold;", cveth.R, cveth.G, cveth.B);
                writer.Write("}\n");
                writer.Write("</style>\n");
                writer.Write("</head>\n");
                writer.Write("<body>\n");
                writer.Write("<center>\n");
                writer.Write("<p class=zag align=center>\n{0}\n</p>\n", zaglav);
writer.Write("<table width=90% border=1 bordercolor=#000000 cellspacing=0 cellpadding=0
valign=center style='width:90.0%;border-collapse:collapse;border:none;'>\n");
                kol_p = dt.Table.Columns.Count;
                writer.Write("<tr>\n");
                //Експорт заголовків стовпців
                for (i = 0; i < kol_p; i++)
                {
 vt = string.Format("    <td class=shapka width={0:F2}% align=center>", 100.0 / kol_p);
                    vt = vt.Replace(',', '.');
                    writer.Write(vt);
                    writer.Write(dt.Table.Columns[i].ColumnName);
                    writer.Write("</td>\n");
                }
                writer.Write("</tr>\n");
                //Експорт даних
                foreach (DataRowView row in dt)
                {
                    writer.Write("<tr>\n");
                    for (i = 0; i < kol_p; i++)
                    {
                        writer.Write("    <td class=text align=center>");
                        writer.Write(" ");
                        vt = row[i].ToString();
                     if (dt.Table.Columns[i].DataType.ToString() == "System.DateTime")
                            vt = Convert.ToDateTime(row[i]).ToShortDateString();
                        if (vt != "")
                        {
                            writer.Write(vt);
                            writer.Write(" ");
                        }
                        writer.Write("</td>\n");
                    }
                    writer.Write("</tr>\n");
                }
                writer.Write("</table>\n<br>\n");
                writer.Write("\n");
                writer.Write("</body>\n");
                writer.Write("</html>\n");
                writer.Close();
                return true;
            }
            catch (Exception)
            {
                return false;
            }
        }

        //Користувацькі відгуки
        //Інформація по туристам
    private void информацияПоТуристамToolStripMenuItem_Click(object sender, EventArgs e)
```

```
{
    Form2 ff = new Form2();
    ff.conn = conn;
    ff.ShowDialog();
}
//Готелі
private void отелиToolStripMenuItem_Click(object sender, EventArgs e)
{
    Form4 ff = new Form4();
    ff.conn = conn;
    if(ff.ShowDialog()==DialogResult.OK)
    {
        //Виклик форми з графікою
        Form5 f = new Form5();
        f.conn = conn;
        f.kod_otel = ((Name_List)ff.comboBox3.SelectedItem).ItemData;
        f.ShowDialog();
    }
}
//Доступні готелі
private void доступныеОтелиToolStripMenuItem_Click(object sender, EventArgs e)
{
    Form6 ff = new Form6();
    ff.conn = conn;
    if (ff.ShowDialog() == DialogResult.OK)
    {
        //Виклик форми з графікою
        Form7 f = new Form7();
        f.conn = conn;
        f.kod_gorod = ((Name_List)ff.comboBox2.SelectedItem).ItemData;
        f.ShowDialog();
    }
}
//Вартість туру
private void стоимостьТураToolStripMenuItem_Click(object sender, EventArgs e)
{
    Form8 ff = new Form8();
    ff.conn = conn;
    ff.st = st;
    if (ff.ShowDialog() == DialogResult.OK)
    {
        st = ff.st;
        if (ff.checkBox2.Checked)
        {
            //Виведення інформації
            Form9 f1 = new Form9();
            f1.st = st;
            f1.ShowDialog();
        }
    }
}
//Робота з країнами
private void страныToolStripMenuItem_Click(object sender, EventArgs e)
{
    FormCountry ff = new FormCountry();
    ff.conn = conn;
    ff.ShowDialog();
}
//Робота з містами
private void городаToolStripMenuItem_Click(object sender, EventArgs e)
```

```
        {
            FormCity ff = new FormCity();
            ff.conn = conn;
            ff.ShowDialog();
        }
        //Підключення 3D-графіки через C++ (процес)
        private void dToolStripMenuItem_Click(object sender, EventArgs e)
        {
            try
            {
                System.Diagnostics.Process.Start("Gr_3d.exe");
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }
        //Вибір шрифту
        private void выборШрифтаToolStripMenuItem_Click(object sender, EventArgs e)
        {
            fontDialog1.Font = font_text;
            if (fontDialog1.ShowDialog() == DialogResult.OK)
            {
                font_text = fontDialog1.Font;
                kod_zap = 1;
            }
        }
        //Колір заголовку таблиці
    private void цветЗаголовкаТаблицыToolStripMenuItem_Click(object sender, EventArgs e)
        {
            colorDialog1.Color = cvet_zag;
            if (colorDialog1.ShowDialog() == DialogResult.OK)
            {
                cvet_zag = colorDialog1.Color;
                kod_zap = 1;
            }
        }
        //Колір заголовків стовпців таблиці
       private void цветШапкиТаблицыToolStripMenuItem_Click(object sender, EventArgs e)
        {
            colorDialog1.Color = cvet_hapka;
            if (colorDialog1.ShowDialog() == DialogResult.OK)
            {
                cvet_hapka = colorDialog1.Color;
                kod_zap = 1;
            }
        }
        //Колір тексту таблиці
      private void цветТекстаТаблицыToolStripMenuItem_Click(object sender, EventArgs e)
        {
            colorDialog1.Color = cvet_text;
            if (colorDialog1.ShowDialog() == DialogResult.OK)
            {
                cvet_text = colorDialog1.Color;
                kod_zap = 1;
            }
        }
        //Гістограма в HTML
        private void гістограмаВHTMLToolStripMenuItem_Click(object sender, EventArgs e)
        {
```

```
            //Формування масивів для побудови діаграми
            List<string> infoList = new List<string>();
            List<double> cenaList = new List<double>();
            // SQL запит
            string query = "SELECT Name, Stoim_sytki From OTEL";
            OleDbCommand command = new OleDbCommand(query, conn);
            // Виконання запиту
            OleDbDataReader reader = command.ExecuteReader();
            int i = 0, kol = 10;
            while (reader.Read())
            {
                i++;
                if (i > kol) break;
                // Додаємо дані до списків
                infoList.Add(reader["Name"].ToString());
                cenaList.Add(Convert.ToDouble(reader["Stoim_sytki"]));
            }
            reader.Close();
            command.Dispose();
            //Виклик скрипту для формування HTML
            SaveHtmlFile("Diagram.htm", "Гістограма ціни готелей",
                "Ціна готелей за добу (грн.)", infoList.ToArray(), cenaList.ToArray());
            // Запускаємо браузер за умовчанням для відкриття HTML файлу
         Process.Start(new ProcessStartInfo("Diagram.htm") { UseShellExecute = true });
        }
        //Допомога по пакету
        private void помощьToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Help.ShowHelp(null, "turist.chm", "main.htm");
        }
        //Про програму
        private void оПрограммеToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Form10 ff = new Form10();
            ff.ShowDialog();
        }
        protected override void OnClosing(CancelEventArgs e)
        {
            if (kod_zap == 1)
            {
                DialogResult dr = MessageBox.Show("Налаштування були змінені!\nЗберегти
зміни на диск?", "Зберігання",
                MessageBoxButtons.YesNoCancel, MessageBoxIcon.Question);
                if (dr == DialogResult.Yes)
                {
                BinaryWriter bb = new BinaryWriter(new FileStream("settings.ini",
FileMode.OpenOrCreate));
                    bb.Write(cvet_zag.R);
                    bb.Write(cvet_zag.G);
                    bb.Write(cvet_zag.B);
                    bb.Write(cvet_hapka.R);
                    bb.Write(cvet_hapka.G);
                    bb.Write(cvet_hapka.B);
                    bb.Write(cvet_text.R);
                    bb.Write(cvet_text.G);
                    bb.Write(cvet_text.B);
                    bb.Write(font_text.Name);
                    bb.Write(font_text.SizeInPoints);
                    bb.Write((int)font_text.Style);
                    bb.Close();
```

```
                    return;
                }
                else if (dr == DialogResult.Cancel)
                    e.Cancel = true;
            }
            base.OnClosing(e);
        }

    }//end Form
    //Клас для заповнення списку
    public class Name_List
    {
        public string str;
        public int ItemData;
        public Name_List(string str, int ItemData)
        {
            this.str = str;
            this.ItemData = ItemData;
        }
        public override string ToString()
        {
            return str;
        }
    }
    //Клас налаштувань для розрахунку вартості (для Form8)
    public class stoim
    {
        public string otel_str;
        public double cena;
        public double skidka;
        public double cena_skidka;
        public int kol_mest;
        public int otel;
        public int sost;
        public int val=0;
        public double grn = 38;
        public double euro = 0.9;//29.6;
        public double cena_dor;
        public double oplata;
        public double predoplata;
        public double ostatok;
        public double stoimost;
        public int chel;
        public int deti;
        public int dni;
        public int dop;
    }
}
```

Додаток Б

Лістинг «Form2»

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.OleDb;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
namespace Tyrfirma
{
    public partial class Form2 : Form
    {
        public OleDbConnection conn;
        OleDbDataAdapter da = null;
        OleDbCommandBuilder bulder = null;
        DataSet ds;
        //Конструктор
        public Form2()
        {
            InitializeComponent();
            ds = new DataSet();
        }
        //Клієнти
        private void button1_Click(object sender, EventArgs e)
        {
                string str = "SELECT Klienti.Kod_klienta AS Код, Klienti.Familija AS
Прізвище, Klienti.Imia AS [Ім'я], Klienti.Viza AS Віза, Klienti.God_rogd AS
Дата_народження, Klienti.Zarplata AS Зарплатня FROM Klienti";
                Zapros(str,"Клієнти");
                label1.Text = "Клієнти";
        }
        //Готелі
        private void button2_Click(object sender, EventArgs e)
        {
                string str = "SELECT Strana.Strana AS Країна, Goroda.Gorod AS Місто,
Otel.Name AS Назва, Otel.Otel_zvezd AS Клас, Otel.Kolich_mest AS Місця,
Otel.Kolich_ekskyrs AS Екскурсії, Otel.Pitanie_raz AS Харчування, Otel.Stoim_sytki AS
Вартість, Otel.Basein AS Басейн, Otel.Restoran AS Ресторан FROM (Strana INNER JOIN
Goroda ON Strana.Kod = Goroda.Kod_str) INNER JOIN Otel ON Goroda.Kod = Otel.Kod_gor";
                Zapros(str,"Готелі");
                label1.Text = "Готелі";
        }
        //Договори на тури
        private void button3_Click(object sender, EventArgs e)
        {
                string str = "SELECT Dogovor.Nom_dogovora, Dogovor.Data_nachala_tyra,
Dogovor.Data_okonch_tyra, Strana.Strana, Goroda.Gorod, Tyri.Stoimost, Tyri.Skidka,
Tyri.Otel, Tyri.Stadija_vipoln, Tyri.Stoim_bileta, Tyri.Tip_nomera_kod,
Tyri.Vid_nomera_mest, Tip_nomera.Dopoln FROM Klienti INNER JOIN ((Tip_nomera INNER JOIN
(Strana INNER JOIN (Goroda INNER JOIN (Tyri INNER JOIN Dogovor ON
Tyri.Kod_tura=Dogovor.Tyr) ON Goroda.Kod=Tyri.Kod_gor) ON Strana.Kod=Goroda.Kod_str) ON
Tip_nomera.Kod_tip=Tyri.Tip_nomera_kod) INNER JOIN Klienti_po_dogovory ON
Dogovor.Nom_dogovora=Klienti_po_dogovory.nom_dogovor) ON
Klienti.Kod_klienta=Klienti_po_dogovory.Klient";
                Zapros(str,"Договори на тури");
                label1.Text = "Договори на тури";
        }
```

```
        //До оплати клієнту
        private void button4_Click(object sender, EventArgs e)
        {
            string str = "SELECT Tyri.Kod_tura, Tyri.Kod_gor, Tyri.Stoimost,
Tyri.Skidka, Otel.Kod_otel, Otel.Name, Tyri.Stoim_bileta, Otel.Stoim_sytki,
Klienti.Kod_klienta, Klienti.Familija, Klienti.Zarplata, [Tyri]![Stoimost]*(1-
[Tyri]![Skidka]/100) AS [Зі знижкою], IIf([Klienti]![Zarplata]<5000,[Зі
знижкою]*0.9,[Зі знижкою]) AS [До оплати] FROM Klienti, (Goroda INNER JOIN Otel ON
Goroda.Kod = Otel.Kod_gor) INNER JOIN Tyri ON Goroda.Kod = Tyri.Kod_gor";
            Zapros(str,"До оплати клієнту");
            label1.Text = "До оплати клієнту";
        }
        //Функція реалізації запиту
        private void Zapros(string str,string name)
        {
            try
            {
                if (da != null) da.Dispose();
                if (bulder != null) bulder.Dispose();
                da = new OleDbDataAdapter(str, conn);
                bulder = new OleDbCommandBuilder(da);
                bulder.QuoteSuffix = "]";
                bulder.QuotePrefix = "[";
                ds.Tables.Clear();
                da.Fill(ds,name);
                bindingSource1.DataSource = ds.Tables[0];
                dataGridView1.DataSource = bindingSource1;
                bindingNavigator1.BindingSource = bindingSource1;
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }
        //Кнопка оновити
        private void toolStripLabel1_Click(object sender, EventArgs e)
        {
            if (dataGridView1.RowCount == 0) return;
            try
            {
                da.Update(ds.Tables[0]);
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }
        //Експорт у XLS
        private void toolStripButton1_Click(object sender, EventArgs e)
        {
            if (dataGridView1.RowCount == 0) return;
            if (!Form1.ExportXLS(ds.Tables[0].DefaultView))
            {
                MessageBox.Show("Неможливий експорт в Excel!");
            }
        }
        //Експорт у HTML
        private void toolStripButton2_Click(object sender, EventArgs e)
        {
            if (dataGridView1.RowCount == 0) return;
```

Продовження додатку Б

```
                if (!Form1.Export_Html(ds.Tables[0].TableName + ".htm",
ds.Tables[0].DefaultView, ds.Tables[0].TableName,
                Form1.font_text, Form1.cvet_zag, Form1.cvet_hapka, Form1.cvet_text))
            {
                MessageBox.Show("Неможливий експорт в HTML!");
                return;
            }
            Form3 ff = new Form3();
      ff.webBrowser1.Navigate(Application.StartupPath + "\\" + ds.Tables[0].TableName +
".htm");
            ff.ShowDialog();
        }
    }
}
```

Додаток В

Лістинг «Form3»

```
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 Tyrfirma
{
    public partial class Form3 : Form
    {
        public Form3()
        {
            InitializeComponent();
        }
    }
}
```

Додаток Г

Лістинг «Form4»

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.OleDb;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace Tyrfirma
{
    public partial class Form4 : Form
    {
        public OleDbConnection conn;

        public Form4()
        {
            InitializeComponent();
        }
        //Завантаження форми
        private void Form4_Load(object sender, EventArgs e)
        {
            try
            {
           OleDbCommand cmd = new OleDbCommand("SELECT Kod,Strana From Strana", conn);
                OleDbDataReader reader = cmd.ExecuteReader();

                //Зчитування даних у список
                while (reader.Read())
              comboBox1.Items.Add(new Name_List(reader[1].ToString(), (int)reader[0]));
                reader.Close();
                cmd.Dispose();
                if(comboBox1.Items.Count>0)comboBox1.SelectedIndex = 0;
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }
        //Вибір міста по країні
        private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
        {
            try
            {
                if (comboBox1.Items.Count == 0) return;
                string str = "SELECT Kod,Gorod From Goroda WHERE Kod_str=";
                str += ((Name_List)comboBox1.SelectedItem).ItemData.ToString();
                OleDbCommand cmd = new OleDbCommand(str, conn);
                OleDbDataReader reader = cmd.ExecuteReader();
                //Очищення даних списку
                comboBox2.Items.Clear();
                //Зчитування даних у список
                while (reader.Read())
             comboBox2.Items.Add(new Name_List(reader[1].ToString(), (int)reader[0]));
               reader.Close();
                cmd.Dispose();
                if (comboBox2.Items.Count > 0) comboBox2.SelectedIndex = 0;
```

```
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }
        //Вибір готелю по місту
        private void comboBox2_SelectedIndexChanged(object sender, EventArgs e)
        {
            try
            {
                if (comboBox2.Items.Count == 0) return;
                string str = "SELECT Kod_otel,Name From Otel WHERE Kod_gor=";
                str += ((Name_List)comboBox2.SelectedItem).ItemData.ToString();
                OleDbCommand cmd = new OleDbCommand(str, conn);
                OleDbDataReader reader = cmd.ExecuteReader();
                //Очищення даних списку
                comboBox3.Items.Clear();
                //Зчитування даних у список
                while (reader.Read())
             comboBox3.Items.Add(new Name_List(reader[1].ToString(), (int)reader[0]));
                reader.Close();
                cmd.Dispose();
                if (comboBox3.Items.Count > 0) comboBox3.SelectedIndex = 0;
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }
        //Кнопка Ок
        private void button1_Click(object sender, EventArgs e)
        {
            if (comboBox3.Items.Count == 0) return;
            DialogResult = DialogResult.OK;
        }
        //Кнопка Відміна
        private void button2_Click(object sender, EventArgs e)
        {
            DialogResult = DialogResult.Cancel;
        }
    }
}
```

Додаток Д

Лістинг «Form5»

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.OleDb;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
namespace Tyrfirma
{
    //Клас для виведення інформації
    public partial class Form5 : Form
    {
        public OleDbConnection conn;
        //Змінна коду готеля
        public int kod_otel = 0;
        //Масив даних для графіки
        double[] mas_dan;
        string[] mas_viv;
       //Масив кольорів для виведення гістограми
        Color[] mascolor =
        {
            Color.Blue,Color.Green,
            Color.Red,Color.Yellow,Color.Tomato,Color.Silver,
            Color.SeaGreen,Color.SandyBrown,Color.SeaShell,Color.SlateBlue,
            Color.Violet,Color.YellowGreen,Color.Teal,Color.SkyBlue,
        };
        //Функція роботи з двувимірною графікою
        //Коментар для гістограм
        /*
        mas - вказівка на масив входу;
        rect - прямокутник виведення
        gr - об'єкт Graphics;
        kod - характеристика виведення написів
            0 - виведення написів (за умовчанням)
            1 - відсутність написів
        dec - кількість знаків після коми при виведенні (за умовчанням 1)
        kl - ключ виведення інформації
            0 - виведення значень масиву (за умовчанням)
            1 - виведення долі у %-х
        mascolor - масив кольорів Color
        */
        void Bar(double[] mas, ref Rectangle rect, Graphics gr, int kod,
            int dec, int kl, Color[] mascolor)
        {
            int i, kol_sim, h, n;
            double proc, sum, max;
            double otstup, ch_st, dop;
            double k_ed_h, delta;
            int nad, proc_i, proc_imax;
            //proc_imax - верхня ціна ділення для цілих чисел
            string dd, str;
            SizeF pram, pram1, vrr;
            Point p2;
            Rectangle rr1;
            Brush br;
            sum = 0;
```

```
        max = mas[0];
        n = mas.Length;
        //Пошук суми елементів та максимуму
        for (i = 0; i < n; i++)
        {
            if (mas[i] > max) max = mas[i];
            sum += mas[i];
        }
        if (sum == 0) return;
        if (kl == 1) max = 100.0 * max / sum;
        //Формування шаблону відносно кількості цифр
        dd = string.Format("{0}", dec);
        dd = "{0:F" + dd + "}";
        str = string.Format("{0:F0}", max);
        kol_sim = str.Length;
        Font font = new Font("Times New Roman", 10, FontStyle.Bold);
        pram = gr.MeasureString(str, font);
        otstup = pram.Width + 2.0 * pram.Width / kol_sim;
        ch_st = (rect.Width - otstup) / (2.0 * n + 1);
        Random rnd = new Random(48);
        h = rect.Height;
        //Знаходження максимуму типа int для вісі Y
        delta = (kl == 0) ? sum - max : 100.0 - max;
        proc_i = (dec == 0) ? (int)max : (int)Math.Ceiling(max);
        proc_imax = proc_i;
        if (Math.Abs(delta) > 0.00001 && proc_i % 9 > 0)
            proc_imax = (proc_i / 9 + 1) * 9;
        //Кількість одиниць на висоту
        k_ed_h = 0.9 * h / proc_imax;
        pram1 = gr.MeasureString("9", font);
        br = new SolidBrush(Color.Black);
        Pen pen = new Pen(Color.Black);
        for (i = 0; i < n; i++)
        {
            proc = (kl == 1) ? 100.0 * mas[i] / sum : mas[i];
            dop = (proc < max / 100.0) ? max / 100.0 : proc;
            Point p1 = new Point((int)(otstup + ch_st * (2 * i + 1) + rect.Left),
                      (int)(0.95 * h - dop * k_ed_h + rect.Top));
            p2 = p1;
            p2.Offset((int)ch_st, (int)(dop * k_ed_h));
            if (mascolor == null)
br = new SolidBrush(Color.FromArgb(rnd.Next(256), rnd.Next(256), rnd.Next(256)));
            else
                br = new SolidBrush(mascolor[i % 13]);
            rr1 = new Rectangle(p1.X, p1.Y, p2.X - p1.X, p2.Y - p1.Y);
            gr.FillRectangle(br, rr1);
            gr.DrawRectangle(pen, rr1);
            str = string.Format(dd, proc);
            vrr = gr.MeasureString(str, font);
            nad = (int)vrr.Width + 4;
            if (nad <= 2 * ch_st)//
            {
                if (p2.Y - p1.Y >= pram1.Height * 1.2 && ch_st >= nad && kod == 0)
                {
                    StringFormat ss = new StringFormat();
                    ss.Alignment = StringAlignment.Center;
                    ss.LineAlignment = StringAlignment.Center;
                    gr.DrawString(str, font, new SolidBrush(Color.Black), rr1, ss);
                }
                else
```

```
                        if (p1.Y - rect.Top >= 1.2 * pram1.Height && kod == 0)
                        {
       rr1 = new Rectangle((int)(p1.X - 0.5 * ch_st), (int)(p1.Y - 1.2 * pram1.Height),
                                (int)(2 * ch_st), (int)(1.2 * pram1.Height));
                            StringFormat ss = new StringFormat();
                            ss.Alignment = StringAlignment.Center;
                            ss.LineAlignment = StringAlignment.Center;
                       gr.DrawString(str, font, new SolidBrush(Color.Black), rr1, ss);
                        }
                } // зат if
            } // for
            gr.DrawLine(pen, (int)(otstup + rect.Left), (int)(0.05 * h + rect.Top),
                        (int)(otstup + rect.Left), (int)(0.95 * h + rect.Top));
            gr.DrawLine(pen, (int)(otstup + rect.Left), (int)(0.95 * h + rect.Top),
          (int)(otstup + rect.Left + ch_st * (2 * n + 1)), (int)(0.95 * h + rect.Top));
            if (Math.Abs(delta) < 0.00001)
            {
                str = string.Format("{0}", proc_imax);
                gr.DrawString(str, font, new SolidBrush(Color.Black),
      rect.Left + pram.Width / kol_sim, (int)(rect.Top + 0.05 * h - pram.Height / 2));
                for (i = 1; i <= 9; i++)
                {
                    gr.DrawLine(pen, (int)(rect.Left + otstup), (int)(0.1 * h * (i - 1)
+ rect.Top + 0.05 * h),
                        (int)(rect.Left + otstup + 0.3 * ch_st), (int)(0.1 * h * (i -
1) + rect.Top + 0.05 * h));
                }//end for
            }
            else
            {
                for (i = 1; i <= 9; i++)
                {
                    gr.DrawLine(pen, (int)(rect.Left + otstup), (int)(0.1 * h * (i - 1)
+ rect.Top + 0.05 * h),
                        (int)(rect.Left + otstup + 0.3 * ch_st), (int)(0.1 * h * (i -
1) + rect.Top + 0.05 * h));
                    if (i == 1 && kl == 1 && proc_imax > 100) continue;
                 str = string.Format("{0:F0}", proc_imax * (1.0 - (i - 1) * 1.0 / 9));
                    if ((i & 1) != 0)
                        gr.DrawString(str, font, new SolidBrush(Color.Black),
                           rect.Left + pram.Width / kol_sim, (int)(rect.Top + 0.1 * h *
(i - 1) + 0.05 * h - pram.Height / 2));
                }//end for
            }//end if
        }
        public Form5()
        {
            InitializeComponent();
        }

        //Загрузка формы
        private void Form5_Load(object sender, EventArgs e)
        {
            try
            {
             OleDbCommand cmd = new OleDbCommand("SELECT
Kod_otel,Kod_gor,Name,Otel_zvezd,Stoim_sytki From OTEL", conn);
                OleDbDataReader reader = cmd.ExecuteReader();
                int i, kol = 10;
                //Зчитування даних у масив
```

```
                mas_dan = new double[kol];
                for (i = 0; i < kol; i++)
                {
                    reader.Read();
                    mas_dan[i] = Convert.ToDouble(reader[4]);
                }
                reader.Close();
                cmd.Dispose();
                //Заповнення масиву для виведення
                mas_viv = new string[7];
                string str = "SELECT * FROM OTEL WHERE Kod_otel=";
                str += kod_otel;
                cmd = new OleDbCommand(str, conn);
                reader = cmd.ExecuteReader();
                reader.Read();
                mas_viv[0] = reader[12].ToString();
                for (i = 2; i <= 7; i++)
                    mas_viv[i - 1] = reader[i].ToString();
                reader.Close();
                cmd.Dispose();
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }
        //Функція виведення інформації (малювання)
        private void Form5_Paint(object sender, PaintEventArgs e)
        {
            Graphics gr = e.Graphics;
            //Завантаження картинки
            string str = string.Format("BMP\\otel{0}.bmp",mas_viv[0]);
            Image img = Image.FromFile(str);
            gr.DrawImage(img, 10, 10);
            float koef = 1.5f;
            float vis,ots_x = 380;
            vis = gr.MeasureString("W", Form1.font_text).Height;

            //Виведення інформації
            str = "Назва готелю:   ";
            str += mas_viv[1];
gr.DrawString(str, Form1.font_text, new SolidBrush(Form1.cvet_text),ots_x, koef * vis);
            koef += 1.5f;
            str = "Зірковість:   ";
            str += mas_viv[2];
gr.DrawString(str, Form1.font_text, new SolidBrush(Form1.cvet_text), ots_x, koef * vis);
            koef += 1.5f;
            str = "Кількість місць:   ";
            str += mas_viv[3];
gr.DrawString(str, Form1.font_text, new SolidBrush(Form1.cvet_text), ots_x, koef * vis);
            koef += 1.5f;
            str = "Кількість екскурсій:   ";
            str += mas_viv[4];
gr.DrawString(str, Form1.font_text, new SolidBrush(Form1.cvet_text), ots_x, koef * vis);
            koef += 1.5f;
            str = "Харчування(раз на добу):   ";
            str += mas_viv[5];
gr.DrawString(str, Form1.font_text, new SolidBrush(Form1.cvet_text), ots_x, koef * vis);
            koef += 1.5f;
            str = "Вартість доби проживання:   ";
```

```
            str += mas_viv[6];
gr.DrawString(str, Form1.font_text, new SolidBrush(Form1.cvet_text), ots_x, koef * vis);
            //Виведення діаграми
            Rectangle rect = ClientRectangle;
            Rectangle rect1 =
Rectangle.FromLTRB((int)(1/20.0f*rect.Height),(int)(5/10.0f*rect.Height),
                (int)(rect.Width-4/10.0f*rect.Height),(int)rect.Height);
       //rect = rect.Inflate(((int)(-1/20.0f*rect.Height),(int)(-5/10.0f*rect.Height),
                   //      (int)(-4/10.0f*rect.Height),0);
            Bar(mas_dan, ref rect1, gr, 0, 1, 0, mascolor);
        }
    }
}
```

Додаток Е

Лістинг «Form6»

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.OleDb;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace Tyrfirma
{
    public partial class Form6 : Form
    {
        public OleDbConnection conn;
        public Form6()
        {
            InitializeComponent();
        }
        private void Form6_Load(object sender, EventArgs e)
        {
            try
            {
           OleDbCommand cmd = new OleDbCommand("SELECT Kod,Strana From Strana", conn);
                OleDbDataReader reader = cmd.ExecuteReader();

                while (reader.Read())
              comboBox1.Items.Add(new Name_List(reader[1].ToString(), (int)reader[0]));
                reader.Close();
                cmd.Dispose();
                if (comboBox1.Items.Count > 0) comboBox1.SelectedIndex = 0;
                for (int i = 1; i <= 10; i++)
                {
                    comboBox3.Items.Add(100 * i);
                    comboBox4.Items.Add(100 * i);
                }

                comboBox3.SelectedIndex = 0;
                comboBox4.SelectedIndex = 9;
                comboBox5.Items.Add("1");
                comboBox5.Items.Add("2");
                comboBox5.Items.Add("3");
                comboBox5.SelectedIndex = 0;
             cmd = new OleDbCommand("SELECT Kod_tip,Tip_nomera From Tip_nomera", conn);
                reader = cmd.ExecuteReader();

                while (reader.Read())
              comboBox6.Items.Add(new Name_List(reader[1].ToString(), (int)reader[0]));
                reader.Close();
                cmd.Dispose();
                if (comboBox6.Items.Count > 0) comboBox6.SelectedIndex = 0;
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }
        private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
```

```
        {
            try
            {
                if (comboBox1.Items.Count == 0) return;
                string str = "SELECT Kod,Gorod From Goroda WHERE Kod_str=";
                str += ((Name_List)comboBox1.SelectedItem).ItemData.ToString();
                OleDbCommand cmd = new OleDbCommand(str, conn);
                OleDbDataReader reader = cmd.ExecuteReader();

                //Очищення даних списку
                comboBox2.Items.Clear();
                //Зчитування даних у список
                while (reader.Read())
      comboBox2.Items.Add(new Name_List(reader[1].ToString(), (int)reader[0]));
                reader.Close();
                cmd.Dispose();
                if (comboBox2.Items.Count > 0) comboBox2.SelectedIndex = 0;
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }
        //Обробка запиту
        private void button1_Click(object sender, EventArgs e)
        {
            if (comboBox2.Items.Count == 0)
            {
                MessageBox.Show("Помилка! Оберіть місто");
                return;
            }
            string str = "SELECT
Kod_otel,Kod_gor,Name,Otel_zvezd,Stoim_sytki,Basein,Sobst_bereg_linija,Restoran,Minibar
,TV,Internet,Fitness_zal,Obmen_valut,Dajving_center,Room_service,Sayna,Biznes_zal,Spa_c
enter From OTEL WHERE Kod_gor=";
            str += ((Name_List)comboBox2.SelectedItem).ItemData;

            DialogResult = DialogResult.OK;
        }
        //Кнопка Відміна
        private void button2_Click(object sender, EventArgs e)
        {
            DialogResult = DialogResult.Cancel;
        }
    }
}
```

Додаток Ж

Лістинг «Form7»

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.OleDb;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
namespace Tyrfirma{
    public partial class Form7 : Form
    {
        public OleDbConnection conn;
        public int kod_gorod;
        DataSet ds;
        int pol;
        int[] mas_shir;
        Font font;
        int shir, vis;
        public Form7()
        {
            InitializeComponent();
            pol = 2;
            mas_shir = new int[pol];
            font = Form1.font_text;
            //Визначення ширини у пікселях
            //Створення контексту gr
            Graphics gr = CreateGraphics();
            SizeF s = gr.MeasureString("9", font);
            shir = (int)s.Width;
            vis = (int)s.Height;
            gr.Dispose();
            mas_shir[0] = 20 * shir;
            mas_shir[1] = 8 * shir;
        }
        // Завантаження форми
       private void Form7_Load(object sender, EventArgs e){
            try{
                string str = "SELECT
Kod_otel,Kod_gor,Name,Otel_zvezd,Stoim_sytki,Basein,Sobst_bereg_linija,Restoran,Minibar
,TV,Internet,Fitness_zal,Obmen_valut,Dajving_center,Room_service,Sayna,Biznes_zal,Spa_c
enter From OTEL WHERE Kod_gor=";
                str += kod_gorod;
                OleDbDataAdapter da = new OleDbDataAdapter(str, conn);
                ds = new DataSet();
                da.Fill(ds);
                Graphics gr = CreateGraphics();
                //Визначення скролінгу для Paint
                //this.AutoScrollMinSize = new Size(50+(int)gr.MeasureString("Сводная
информация о выбранном клиентом отеле", Form1.font_text).Width,
                    //(int)(30*gr.MeasureString("W",
Form1.font_text).Height*ds.Tables[0].Rows.Count));
                this.AutoScroll = true;
                panel1.Size = new Size(50+(int)gr.MeasureString("Зведена  інформація
про  вибраний  клієнтом  готель", Form1.font_text).Width,
                    (int)(37*gr.MeasureString("W",
Form1.font_text).Height*ds.Tables[0].Rows.Count));
                gr.Dispose();
```

```
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }

        //Малювання у панелі
        private void panel1_Paint(object sender, PaintEventArgs e)
        {
            Graphics gr = e.Graphics;
            float koef = 1.5f;
            float x,y, vis, ots_x = 50;
            Point pt = Point.Empty;
            RectangleF r1;
            string s = "";
            vis = gr.MeasureString("W", Form1.font_text).Height;
           string str = "Зведена інформація про вибраний клієнтом готель";
      gr.DrawString(str, Form1.font_text, Brushes.Red, pt.X + ots_x, pt.Y + koef * vis);
            koef += 3f;
            y = koef * vis;
            //Масив для виведення
            string[] str1 = {
             "Назва готелю: ","Ранг готелю: ","Вартість доби: ",
             "Наявність басейну: ","Наявність власної берегової лінії: ",
             "Наявність ресторану: ", "Наявність мінібару: ","Наявність телебачення: ",
             "Наявність інтернет-клубу: ","Наявність фітнес-центру: ",
             "Наявність обміну валют: ", "Наявність дайвінг-центру: ",
             "Обслуговування кімнат 24 години: ","Наявність сауни: ",
             "Наявність конференц-залу: ", "Наявність Spa-центру: "
                            };
            //Данні таблиці
            int i,j,n = str1.Length;
            StringFormat ss = new StringFormat();
            ss.Alignment = StringAlignment.Near;
            ss.LineAlignment = StringAlignment.Center;
            DataRow row = ds.Tables[0].Rows[0];
            for (i = 0; i < n; i++)
            {
                x = ots_x;
                for (j = 0; j < pol; j++)
                {
                    r1 = new RectangleF(pt.X + x, pt.Y + y, mas_shir[j], 2 * vis);
                    gr.DrawRectangle(Pens.Black, r1.X,r1.Y,r1.Width,r1.Height);
                    if (j == 0)
                gr.DrawString(str1[i], font, new SolidBrush(Form1.cvet_text), r1, ss);
                    if (j == 1)
                    {
                        if (i < 3) s = row[i + 2].ToString();
                        else if (Convert.ToInt32(row[i + 2]) != 0) s = "Так";
                        else s = "Ні";
                     gr.DrawString(s, font, new SolidBrush(Form1.cvet_text), r1, ss);
                    }
                    x += mas_shir[j];
                }
                y += 2 * vis;
            }
        }
}
}
```

Додаток И

Лістинг «Form8»

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.OleDb;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
namespace Tyrfirma
{
    public partial class Form8 : Form
    {
        public stoim st;
        public OleDbConnection conn;
        public Form8()
        {
            InitializeComponent();
        }
        //Завантаження форми
        private void Form8_Load(object sender, EventArgs e)
        {
            //Заповнення даних
            try
            {
                //st.grn = 4.8;
                //st.euro = 5.2;
                //st.chel = 1;
                //st.deti = 1;
                //st.dni = 1;
                numericUpDown1.Value = st.chel;
                numericUpDown2.Value = st.deti;
                numericUpDown3.Value = st.dni;
                OleDbCommand cmd = new OleDbCommand("SELECT Otel.Kod_otel,
Otel.Kod_gor, Otel.Name, Otel.Otel_zvezd, Otel.Kolich_mest, Otel.Kolich_ekskyrs,
Otel.Pitanie_raz, Otel.Stoim_sytki FROM Otel", conn);
                OleDbDataReader reader = cmd.ExecuteReader();
                //Зчитування даних у список
                while (reader.Read())
                    combo_otel.Items.Add(new Name_List(reader[2].ToString(),
(int)reader[0]));
                reader.Close();
                cmd.Dispose();
                if (combo_otel.Items.Count > 0) combo_otel.SelectedIndex = 0;
                ////////////////////
                trackBar1.Minimum = 0;
                trackBar1.Maximum = 3;
                trackBar1.Value = st.sost;
                trackBar1.TickFrequency = 1;
                trackBar1.SmallChange = 1;
                trackBar1.LargeChange = 3;
                if (st.val == 0) label16.Text = "Evro";
                if (st.val == 1) label16.Text = "$";
                if (st.val == 2) label16.Text = "грн";
                string str;
                str = string.Format("{0:F2}",st.grn);
                textBox7.Text = str;
                str = string.Format("{0:F2}", st.euro);
```

```
                textBox8.Text = str;
                trackBar1_Scroll(null, null);
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
            ////////////////////////////////////////
        }
        //Вибір готелю
        private void combo_otel_SelectedIndexChanged(object sender, EventArgs e)
        {
            if (combo_otel.Items.Count==0) return;
            try
            {
                OleDbCommand cmd = new OleDbCommand("SELECT Tip_nomera.Kod_tip,
Tip_nomera.Tip_nomera FROM Tip_nomera", conn);
                OleDbDataReader reader = cmd.ExecuteReader();
                combo_tip.Items.Clear();
                //Зчитування даних у список
                while (reader.Read())
       combo_tip.Items.Add(new Name_List(reader[1].ToString(), (int)reader[0]));
                reader.Close();
                cmd.Dispose();
                if (combo_tip.Items.Count > 0) combo_tip.SelectedIndex = 0;
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }
        //Вибір типу номеру
        private void combo_tip_SelectedIndexChanged(object sender, EventArgs e)
        {
            if (combo_tip.Items.Count == 0) return;
            try
            {
                OleDbCommand cmd = new OleDbCommand("SELECT Vid_nomera_mest FROM Tyri
Group By Vid_nomera_mest Order By Vid_nomera_mest", conn);
                OleDbDataReader reader = cmd.ExecuteReader();
                combo_vid.Items.Clear();
                //Зчитування даних у список
                while (reader.Read())
                    combo_vid.Items.Add(reader[0].ToString());
                reader.Close();
                cmd.Dispose();
                if (combo_vid.Items.Count > 0) combo_vid.SelectedIndex = 0;
                ////////////////////////////////////////////////////////
                string str = "SELECT Dopoln FROM Tip_nomera WHERE Kod_tip=";
                str += ((Name_List)combo_tip.SelectedItem).ItemData;
                cmd = new OleDbCommand(str, conn);
                reader = cmd.ExecuteReader();
                reader.Read();
                //Додаткова вартість
                dop_st.Text = reader[0].ToString();
                reader.Close();
                cmd.Dispose();
            }
            catch (Exception ex)
            {
```

```
        MessageBox.Show(ex.Message);
    }
}
//Подія на trackBar
private void trackBar1_Scroll(object sender, EventArgs e)
{
    st.sost = trackBar1.Value;
    if (st.sost == 0) sost.Text = "відмінне";
    if (st.sost == 1) sost.Text = "гарне";
    if (st.sost == 2) sost.Text = "середнє";
    if (st.sost == 3) sost.Text = "погане";
    cena();
}
private void button1_Click(object sender, EventArgs e)
{
    st.val = 0;
    label16.Text = "Evro";
    cena();
}
private void button2_Click(object sender, EventArgs e)
{
    st.val = 1;
    label16.Text = "$";
    cena();
}
private void button3_Click(object sender, EventArgs e)
{
    st.val = 2;
    label16.Text = "грн";
    cena();
}
private void checkBox1_CheckedChanged(object sender, EventArgs e)
{
    if (checkBox1.Checked == false) return;
    cena();
}
//Кнопка Ok
private void button4_Click(object sender, EventArgs e)
{
    if (checkBox2.Checked==true)
    {
        // Виведення даних
        // doc->kod_graf = 3;
        // doc->kod_grafik = 0;
        st.otel_str = ((Name_List)combo_otel.SelectedItem).str;
        st.cena = double.Parse(textBox1.Text);
        st.oplata = double.Parse(textBox3.Text);
        st.predoplata = double.Parse(textBox4.Text);
        st.ostatok = double.Parse(textBox6.Text);
        st.cena_dor = double.Parse(textBox2.Text);
        st.cena_skidka = double.Parse(textBox5.Text);
    }
    DialogResult = DialogResult.OK;
}
//Кнопка Відміна
private void button5_Click(object sender, EventArgs e)
{
    DialogResult = DialogResult.Cancel;
}
//Функція ціни
```

```
        private void cena()
        {
            double m_cena;
            double m_skidka;
            double m_cena_skidka;
            double m_stoimost = 0;
            double m_cena_dor;
            double m_oplata;
            double m_predoplata;
            double m_ostatok;
            ////////////////////////////////////////////////////
            try
            {
                st.chel = (int)numericUpDown1.Value;
                st.deti = (int)numericUpDown2.Value;
                st.dni = (int)numericUpDown3.Value;
                st.dop = int.Parse(dop_st.Text);
                m_cena_skidka = 0;

                //m_cena = 0;
                //m_cena_dor = 0;
                //m_oplata = m_skidka;
                //m_predoplata = doc->options.chel+doc->options.sost;
                //m_ostatok = doc->options.chel+doc->options.sost;
                //m_stoimost = doc->options.chel+doc->options.sost;
                //m_cena = 300;
                string str = "SELECT Stoim_sytki,Skidka FROM OTEL WHERE Kod_otel=";
                str += ((Name_List)combo_otel.SelectedItem).ItemData;
                OleDbCommand cmd = new OleDbCommand(str, conn);
                OleDbDataReader reader = cmd.ExecuteReader();
                reader.Read();
                double dd = Convert.ToDouble(reader[0]);
                m_cena = dd;
                m_skidka = Convert.ToDouble(reader[1]);
                reader.Close();
                cmd.Dispose();
                ////////////////////////////////
                st.cena = m_cena;
                st.skidka = m_skidka;
                m_cena = ((m_cena + st.dop) * st.chel * st.dni) + ((m_cena + st.dop) *
st.deti * st.dni) / 2 + 3 - st.sost;
                m_cena /= 10;
                m_cena_skidka = m_cena - (m_skidka * m_cena / 100) + 3 - st.sost;
                m_cena_dor = dd * 15 / 100 + 3 - st.sost;//*5/100;
                m_cena_dor *= (st.chel + st.deti);
                m_oplata = m_cena_skidka + m_cena_dor + 3 - st.sost;
                m_predoplata = m_cena_skidka * 20 / 100 + m_cena_dor + 3 - st.sost;
                m_ostatok = m_oplata - m_predoplata;
                // m_stoimost = m_cena;
                st.grn = double.Parse(textBox7.Text);
                st.euro = double.Parse(textBox8.Text);
                double per = 1;//доллар
                if (st.val == 0) per = st.euro;
                if (st.val == 2) per = st.grn;
                //Text = st.euro + "  " + st.grn;
                m_cena *= per;
                m_cena_dor *= per;
                m_oplata *= per;
                m_predoplata *= per;
                m_ostatok *= per;
```

```
                m_stoimost *= per;
               m_cena_skidka *= per;
                //  m_cena_skidka = m_cena-(m_skidka*m_cena/100);
                str = string.Format("{0:F2}", m_cena);
                textBox1.Text = str;
                textBox10.Text = str;
                str = string.Format("{0:F2}", m_cena_skidka);
                textBox5.Text = str;
                str = string.Format("{0:F2}", m_cena_dor);
                textBox2.Text = str;
                str = string.Format("{0:F2}", m_oplata);
                textBox3.Text = str;
                str = string.Format("{0:F2}", m_predoplata);
                textBox4.Text = str;
                str = string.Format("{0:F2}", m_ostatok);
                textBox6.Text = str;
            }

            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }
    }
}
```

Додаток К

Лістинг «Form9»

```
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 Tyrfirma
{
    public partial class Form9 : Form
    {
        public stoim st;

        int pol;
        int[] mas_shir;
        Font font;
        int shir, vis;

        public Form9()
        {
            InitializeComponent();
            pol = 2;
            mas_shir = new int[pol];
            font = Form1.font_text;

            //Визначення ширини у пікселях
            //Створення контексту gr
            Graphics gr = CreateGraphics();
            SizeF s = gr.MeasureString("9", font);
            shir = (int)s.Width;
            vis = (int)s.Height;
            gr.Dispose();
            mas_shir[0] = 12 * shir;
            mas_shir[1] = 8 * shir;
        }
        //Завантаження форми
        private void Form9_Load(object sender, EventArgs e)
        {
            Graphics gr = CreateGraphics();
            this.AutoScroll = true;
            panel1.Size = new Size(50 + (int)gr.MeasureString("Розраховані показники
вартості туру", Form1.font_text).Width,
                (int)(22 * gr.MeasureString("W", Form1.font_text).Height));
            gr.Dispose();
        }
        //Виведення інформації у панелі
        private void panel1_Paint(object sender, PaintEventArgs e)
        {
            Graphics gr = e.Graphics;
            float koef = 1.5f;
            float x,y,vis, ots_x = 50;
            Point pt = Point.Empty;
            RectangleF r1;
            vis = gr.MeasureString("W", Form1.font_text).Height;
            string str;
            str = "Розраховані показники вартості туру";
```

```
    gr.DrawString(str, Form1.font_text, Brushes.Red, pt.X + ots_x, pt.Y + koef * vis);
            koef += 3f;
            y = koef * vis;
            //Масиви для виведення
            string[] str1 = {
                                "Назва готелю: ", "Знижка: ","Ціна зі знижкою: ",
                                "Вартість туру: ", "Вартість дороги: ",
                                "До оплати: ", "Передплата: ","Залишок: "
                            };
            string[] str2 = {
                  st.otel_str,st.skidka.ToString(),st.cena_skidka.ToString(),
                  st.cena.ToString(), st.cena_dor.ToString(),
                  st.oplata.ToString(), st.predoplata.ToString(),st.ostatok.ToString()
                            };
            //Данні таблиці
            int i, j, n = str1.Length;
            StringFormat ss = new StringFormat();
            ss.Alignment = StringAlignment.Near;
            ss.LineAlignment = StringAlignment.Center;
            for (i = 0; i < n; i++)
            {
                x = ots_x;
                for (j = 0; j < pol; j++)
                {
                    r1 = new RectangleF(pt.X + x, pt.Y + y, mas_shir[j], 2 * vis);
                    gr.DrawRectangle(Pens.Black, r1.X, r1.Y, r1.Width, r1.Height);
                    if (j == 0)
                 gr.DrawString(str1[i], font, new SolidBrush(Form1.cvet_text), r1, ss);
                    if (j == 1)
                 gr.DrawString(str2[i], font, new SolidBrush(Form1.cvet_text), r1, ss);
                    x += mas_shir[j];
                }
                y += 2 * vis;
            }
        }
    }
}
```

Додаток Л

Лістинг «Form10»

```
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 Tyrfirma
{
    public partial class Form10 : Form
    {
        public Form10()
        {
            InitializeComponent();
        }
        private void button1_Click(object sender, EventArgs e)
        {
            DialogResult = DialogResult.OK;
        }
    }
}
```

Додаток М

Лістинг «FormCity»

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.OleDb;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
namespace Tyrfirma
{
    public partial class FormCity : Form
    {
        public OleDbConnection conn;
        OleDbDataAdapter da=null;
        OleDbCommandBuilder bulder = null;
        DataSet ds = null;

        public FormCity()
        {
            InitializeComponent();
        }
        private void FormCity_Load(object sender, EventArgs e)
        {
            try
            {
                string str = "SELECT Kod, Strana From Strana Order BY Strana";
                OleDbCommand cmd = new OleDbCommand(str, conn);
                OleDbDataReader reader = cmd.ExecuteReader();
                while (reader.Read())
              listBox1.Items.Add(new Name_List(reader[1].ToString(), (int)reader[0]));
                reader.Close();
                cmd.Dispose();
                if (listBox1.Items.Count > 0) listBox1.SelectedIndex = 0;
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }
        //Вибір міст по країні
        private void listBox1_SelectedIndexChanged(object sender, EventArgs e)
        {
            try
            {
                string str = "SELECT Kod, Kod_str, Gorod From Goroda where Kod_str=";
                str += ((Name_List)listBox1.SelectedItem).ItemData;
                str += " Order BY Gorod";
                if (da != null) da.Dispose();
                if (ds != null) ds.Dispose();
                if (bulder != null) bulder.Dispose();
                da = new OleDbDataAdapter(str, conn);
                bulder = new OleDbCommandBuilder(da);
                ds = new DataSet();
                da.Fill(ds);
                //Прив'язка даних
                bindingSource1.DataSource = ds.Tables[0];
                listBox2.DataSource = bindingSource1;
```

```
        listBox2.DisplayMember = "Gorod";
        listBox2_SelectedIndexChanged(null, null);
        textBox1.Enabled = false;
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
}
//Вибір міста
private void listBox2_SelectedIndexChanged(object sender, EventArgs e)
{
    DataRow row = ((DataRowView)(bindingSource1.Current)).Row;
    textBox1.Text = row[2].ToString();
    textBox1.Focus();
    textBox1.SelectionStart = 0;
    textBox1.SelectionLength = 0;
}
//Додати місто
private void checkBox1_CheckedChanged(object sender, EventArgs e)
{
    checkBox2.Enabled = !checkBox1.Checked;
    if (checkBox1.Checked)
    {
        textBox1.Enabled = true;
        textBox1.Text = "";
        listBox1.Enabled = false;
        listBox2.Enabled = false;
    }
    else
    {
        textBox1.Enabled = false;
        DataRow row = ((DataRowView)(bindingSource1.Current)).Row;
        textBox1.Text = row[2].ToString();
        listBox1.Enabled = true;
        listBox2.Enabled = true;
    }
    textBox1.Focus();
}
//Змінити місто
private void checkBox2_CheckedChanged(object sender, EventArgs e)
{
    checkBox1.Enabled = !checkBox2.Checked;
    textBox1.Enabled = checkBox2.Checked;
    textBox1.Focus();
    textBox1.SelectionStart = 0;
    textBox1.SelectionLength = 0;
}
//Зберігання даних
private void button1_Click(object sender, EventArgs e)
{
    string str;
    str = textBox1.Text.Trim();
    if (str == "") return;
    if (checkBox1.Checked) bindingSource1.AddNew();
    DataRow row = ((DataRowView)(bindingSource1.Current)).Row;
    row[2] = str;
    if (checkBox1.Checked)
    {
        row[1] = ((Name_List)listBox1.SelectedItem).ItemData;
```

```
                bindingSource1.MoveNext();
            }
            da.Update(ds.Tables[0]);
            if (checkBox1.Checked)
            {
                listBox1_SelectedIndexChanged(null, null);
                bindingSource1.Position = bindingSource1.Find("Gorod", str);
            }
            if (checkBox1.Checked) textBox1.Text = "";
        }
        //Видалення даних
        private void button2_Click(object sender, EventArgs e)
        {
            bindingSource1.RemoveCurrent();
            da.Update(ds.Tables[0]);
            listBox2_SelectedIndexChanged(null, null);
        }
    }
}
```

Додаток Н

Лістинг «FormCountry»

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.OleDb;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
namespace Tyrfirma
{
    public partial class FormCountry : Form
    {
        public OleDbConnection conn;
        OleDbDataAdapter da = null;
        OleDbCommandBuilder bulder = null;
        DataSet ds = null;
        public FormCountry()
        {
            InitializeComponent();
        }
        private void FormCountry_Load(object sender, EventArgs e)
        {
            Upd();
        }
        //Оновлення даних
        void Upd()
        {
            try
            {
                string str = "SELECT Kod, Strana From Strana Order BY Strana";
                if (da != null) da.Dispose();
                if (ds != null) ds.Dispose();
                if (bulder != null) bulder.Dispose();
                da = new OleDbDataAdapter(str, conn);
                bulder = new OleDbCommandBuilder(da);
                ds = new DataSet();
                da.Fill(ds);
                //Прив'язка даних
                bindingSource1.DataSource = ds.Tables[0];
                listBox1.DataSource = bindingSource1;
                listBox1.DisplayMember = "Strana";
                textBox1.Enabled = false;
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }
        //Вибір країни
        private void listBox1_SelectedIndexChanged(object sender, EventArgs e)
        {
            DataRow row = ((DataRowView)(bindingSource1.Current)).Row;
            textBox1.Text = row[1].ToString();
            textBox1.Focus();
            textBox1.SelectionStart = 0;
            textBox1.SelectionLength = 0;
        }
```

```
        //Додавання країни
        private void checkBox1_CheckedChanged(object sender, EventArgs e)
        {
            checkBox2.Enabled = !checkBox1.Checked;

            if (checkBox1.Checked)
            {
                textBox1.Enabled = true;
                textBox1.Text = "";
                listBox1.Enabled = false;
            }
            else
            {
                textBox1.Enabled = false;
                DataRow row = ((DataRowView)(bindingSource1.Current)).Row;
                textBox1.Text = row[1].ToString();
                listBox1.Enabled = true;
            }
            textBox1.Focus();
        }
        //Зміна країни
        private void checkBox2_CheckedChanged(object sender, EventArgs e)
        {
            checkBox1.Enabled = !checkBox2.Checked;
            textBox1.Enabled = checkBox2.Checked;
            textBox1.Focus();
            textBox1.SelectionStart = 0;
            textBox1.SelectionLength = 0;
        }
        //Зберігання даних
        private void button1_Click(object sender, EventArgs e)
        {
            string str;
            str = textBox1.Text.Trim();
            if (str == "") return;
            if (checkBox1.Checked) bindingSource1.AddNew();
            DataRow row = ((DataRowView)(bindingSource1.Current)).Row;
            row[1] = str;
            if (checkBox1.Checked) bindingSource1.MoveNext();
            da.Update(ds.Tables[0]);
            if (checkBox1.Checked)
            {
                Upd();
                bindingSource1.Position = bindingSource1.Find("Strana", str);
            }
            if (checkBox1.Checked) textBox1.Text = "";
        }
        //Видалення даних
        private void button2_Click(object sender, EventArgs e)
        {
            bindingSource1.RemoveCurrent();
            da.Update(ds.Tables[0]);
            listBox1_SelectedIndexChanged(null, null);
        }
    }
}
```

Додаток О

Лістинг «Program»

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Windows.Forms;

namespace Tyrfirma
{
    static class Program
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Form1());
        }
    }
}
```