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

Золотарьова Максима Олеговича
(прізвище, ім'я, по батькові здобувача)

на тему «Розробка програмного забезпечення для автоматизації формування кулінарних рецептів»
(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 11.06.2025 р. № 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 БАКАЛАВРСЬКУ РОБОТУ

1. Тема роботи «Розробка програмного забезпечення для автоматизації формування кулінарних рецептів»

Керівник роботи к.т.н. Медведєв Д.Г.
затверджені наказом закладу вищої освіти від «04» березня 2025 р. № 222-ст

2. Строк подання здобувачем роботи до «09» 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: кулінарні рецепти

Предмет дослідження: інформаційна підтримка кулінарних рецептів

Мета кваліфікаційної роботи: *розробка програмного забезпечення кулінарних рецептів*

5. Дата видачі завдання «04» березня 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 _____ ___Золотарьов М.О.

(підпис) (прізвище та ініціали)

АНОТАЦІЯ

на кваліфікаційну бакалаврську роботу

«Розробка програмного забезпечення для автоматизації формування кулінарних рецептів»

Золотарьова Максима Олеговича

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, що дозволяє автоматизувати процес формування рецептів з використанням кулінарного довідника та вирішувати наступні задачі:

– ведення даних по стравам;

– ведення даних по продуктам;

– формування рецептів, запис їх до бази даних та автоматичне формування звітів;

– автоматизоване формування звітів та їх експорт в MS Excel та HTML.

Програмне забезпечення розроблено на мові C# з використанням технології ADO .NET для обробки баз даних.

Результати роботи рекомендуються до застосування у кулінарній справі.

Ключові слова: програмне забезпечення, СУБД, ADO .NET, кулінарні рецепти.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

.NET FrameWork	Платформа фірми Microsoft, призначена для розробки нативних та web-додатків
ПЗ	Програмне забезпечення
БД	База даних
СУБД	Система управління базами даних
ADO .NET	ActiveX Data Object для .NET – технологія доступу і управління базами даних для платформи .NET

ЗМІСТ

ВСТУП

Кулінарний довідник – довідкове видання, що містить в основному різні рецепти приготування їжі та напоїв. Сучасні кулінарні довідники, як правило, добре ілюстровані, а крім рецептів можуть містити поради по сервіровці столу, правильному харчуванню, вибору продуктів і кухонної техніки та ін.

У Межиріччі були знайдені древні глиняні таблички, які містили рецепти приготування страв - можливо, це перші кулінарні довідники в історії.

Найдавнішою кулінарною книгою в Європі вважається De re coquinaria, написана в кінці IV - початку V століття на латині. Перше друковане видання цієї книги з'явилося в 1483 році. Цей твір описує давньогрецьку та давньоримську кухні, але при цьому містить мінімум деталей про власне готуванні їжі. Нові європейські кулінарні довідники стали з'являтися в середньовічній Європі лише в XIII столітті, через майже тисячу років.

Зі стародавніх кулінарних книг Китаю популярністю користуються Яншень-фан (бл. 200 р. До н.е..) і Іньшань-чжен'яо («Найважливіші принципи їжі і напоїв»), написана дієтологом Ху Сихуей в 1330 році.

Розквіт кулінарних довідників в Європі і Північній Америці пов'язаний з вікторіанської епохою (2-я половина XIX століття), що надавала величезне значення зовнішньої, обрядовій стороні сімейного життя. Для молодих господарок видавалося безліч посібників по правильному веденню домашнього господарства, як правило, містили і розділи з рецептами страв. В цей час займатися складанням кулінарних книг не гребували навіть іменити літератори, як, наприклад, Олександр Дюма-батько, що видав незадовго до смерті «Великий кулінарний словник», і князь Володимир Одоєвський.

В продовження XX століття кулінарні довідники розвивалися по шляху все більшої спеціалізації. Стали з'являтися кулінарні довідники (книги), розраховані не тільки на широке коло домогосподарок, але і на професіоналів

ресторанного справи, допомоги по кухням народів світу, а також збірники рецептів, відібраних за певним принципом (напр., «200 рецептів страв на відкритому повітрі: гриль, барбекю , шашлик »).

У сучасному світі великим попитом користуються кулінарні довідники, випущені під ім’ям відомого шеф-кухаря, особливо якщо він веде кулінарну передачу на телебаченні. За комерційним успіхом книги, яка пропагує ту чи іншу дієту, як правило, слідує видання збірника рецептів для тих, хто вирішить цієї дієти дотримуватися.

Важливою задачею є підбір програмного забезпечення для кулінарних рецептів.

Мета роботи – розробка програмного забезпечення кулінарних рецептів.

Для поставленої мети треба вирішити наступні завдання:

– ведення даних по стравам;

– ведення даних по продуктам;

– формування рецептів, запис їх до бази даних та автоматичне формування звітів;

– автоматизоване формування звітів та їх експорт в MS Excel та HTML.

Об’єкт роботи – кулінарні рецепти.

Предмет роботи – інформаційна підтримка кулінарних рецептів.

При розробці програмного забезпечення буде використано мову C# у середовищі Microsoft Visual Studio 2019.

РОЗДІЛ 1
ПОСТАНОВКА ЗАДАЧІ

1.1. Характеристика задачі

У даній роботі буде розроблятись програма на мові C# у середовищі Visual Studio 2019, яка буде забезпечувати формування рецептів за допомогою кулінарного довіднику. У ході виконання даної роботи планується розробити наступні класи та функції в них [додаток А]:

1. *Form*1 – головна форма програми;
2. *Form*2 – форма ведення довідників таблиць бази даних: *категорія страв*, *категорія продуктів*, *країни рецептів*, *умови заходів*;
3. *Form*3 – форма ведення даних за реляційним зв’язком один-до-багатьох: *страви*, *продукти*.
4. *Form*4 – форма експорту даних в формат HTML.
5. *Form*5 – форма, призначена для формування рецептів страв та запису їх до бази даних.
6. *Form*6 – форма звітності по обраним рецептам у MS Excel.
7. *Form*7 – інформація про розробника програмного забезпечення.

При розробці програмного забезпечення необхідно створити функціональну схему задачі. У нашому випадку програма повинна виконувати наступні функціональні задачі:

1. Ведення та редагування даних довідників;
2. Ведення та редагування даних страв та продуктів;
3. Формування та виведення звітної інформації.

1.2. Вхідна інформація

До вхідної інформації відносять дані, що необхідні для розв'язання аналітичних задач. Вхідна первинна інформація є найбільш детальною і

становить основу для наступної логічної та арифметичної обробки даних. До вхідної інформації може належати не лише змінна, а й умовно-постійна та постійна інформація за особливо великої ролі умовно-постійної.

До вхідної інформації належить файл бази даних «Кулинария.mdb», який вміщує інформацію по стравам, продуктам, країнам проведення заходів тощо.

1.3. Вихідна інформація

Вихідною інформацією є результати роботи з базою даних. Процес опрацювання даних, що знаходяться в базі або додаються до неї реалізується з використанням алгоритмів розробленої програми.

Ведення бази даних відбувається за допомогою зручних візуалізованих форм, де передбачена робота не тільки з текстовою, а й з графічною інформацією. Уся звітна документація виводиться до пакету MS Excel, а також до формату html.

Розроблене програмне забезпечення дозволить автоматизувати процес розробки кулінарних рецептів та вирішити наступні задачі:

– ведення даних по стравам та їх категоріям;

– ведення даних по продуктам та їх категоріям;

– формування рецептів, запис їх до бази даних та автоматичне формування звітів;

– автоматизоване формування звітів та їх експорт в MS Excel та HTML.

1.4. Аналіз існуючих аналогів програмного забезпечення

Розглянемо огляд програмного забезпечення, яке має можливості кулінарних довідників:

1. ***Електронна кулінарна книга 4.0.***

Це одна з найпоширеніших кулінарних програм. Актуальна версія – 4.0. Програма умовно-безкоштовна. Розробник – HomeSoft group. Сайт програми - www.cooke.ru.

При придбанні повної версії є можливість користуватися великою базою даних кулінарних рецептів і статей, яких близько 24 000. Всі рецепти систематизовані, каталог пророоблений і чіткий (рис. 1.1.).

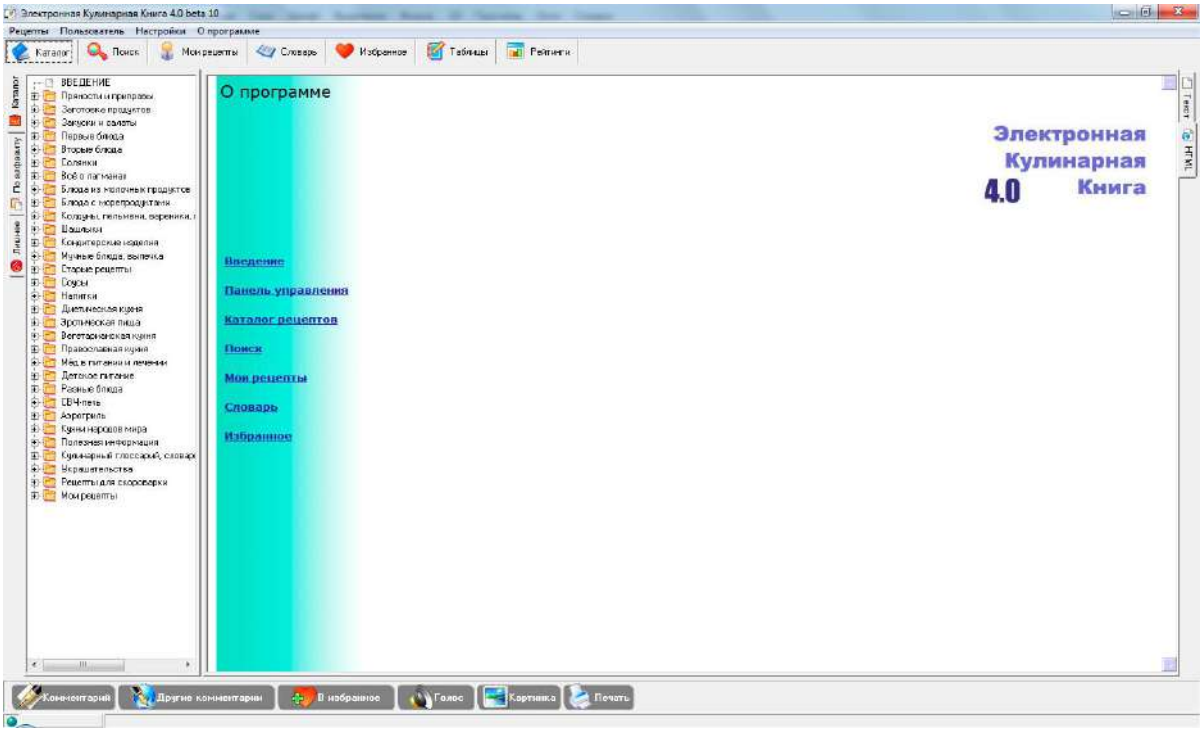

Рис. 1.1. Головне меню програми «Електронна кулінарна книга 4.0»

Є кілька цікавих опцій. Наприклад, система пошуку в програмі дозволяє підбирати страви за інгредієнтами. В цьому випадку щоденне меню можна істотно урізноманітнити.

Програма дозволяє скласти і підтримувати свою власну базу кулінарних рецептів. Рецепти можна оформити на свій розсуд і поданням, додати до них фото і коментарі.

Цікавою задумкою є сервіс корисних порад і кулінарних секретів. Різні нюанси і дрібниці часто можуть істотно оптимізувати домашнє кулінарне виробництво. Будь-який рецепт можна роздрукувати.

З недоліків можна відзначити не дуже зручний редактор для додавання власних рецептів.

Функціональні можливості програми:

- База даних кулінарних рецептів.
- Система пошуку за назвою і за інгредієнтами.
- Особиста кулінарна книга для ведення власних записів.
- Розділ секрети кулінарії з цікавою інформацією про приготування їжі.
- Програма має менеджер звіту друку, який дозволяє вибрати і роздрукувати рецепт або статтю.

2. Книга кулінарних рецептів 6.1.1.

Дана програма має зручний інтерфейс. Актуальна версія програми - 6.1.1. Програма умовно-безкоштовна. Сайт програми - www.lubosoft.ru (рис. 1.2.).

У цій програмі досить зручний інтерфейс. Передбачена можливість змінювати шрифт. Також як і в програмі «Електронна кулінарна книга 4.0» можна здійснювати пошук за інгредієнтами і друкувати рецепти. Але в цій програмі можна зробити вибірку рецептів тільки по одному інгредієнту. Страви, які сподобалися, можна додати в закладки.

В каталозі всього 4 000 страв, каталог невеликий і не деталізований. Непогана підбірка корисних кулінарних порад.

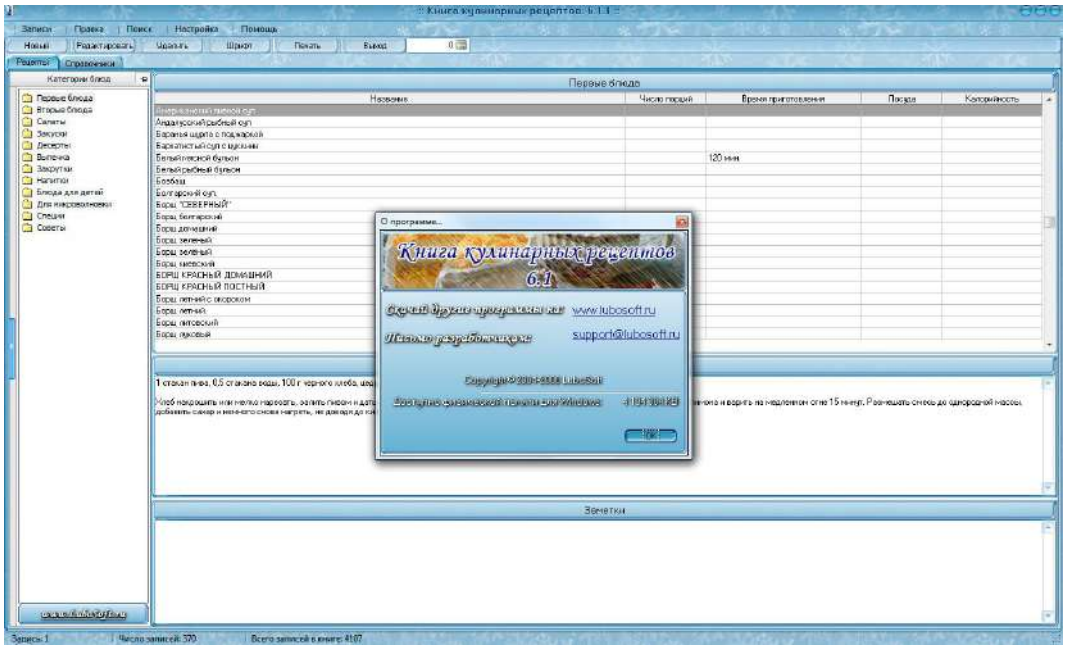

Рис. 1.2. Головне меню програми «Книга кулінарних рецептів 6.1.1.»

Зовнішній вигляд програми можна змінити на свій смак. Розробники пропонують на вибір три скіна для оформлення.

Власна база кулінарних рецептів (її організація) в цій програмі більш продумана. Присутній окремий функціонал для створення категорій і окремий для додавання рецептів.

Функціональні можливості програми:

- Додавання, редагування, видалення рецептів.
- Миттєвий багаторівневий пошук кулінарних рецептів за назвами страв / компонентів (м’ясо, картопля та ін.).
- Додавання вподобаних рецептів в закладки.
- Виділення рецептів кольором.
- Інформаційні довідники, таблиця калорійності продуктів, дієти (в т.ч. Кремлівська дієта з автоматичним підрахунком балів).
- Друк рецептів на принтері.

3. Рецепти 3.0.1.

Досить невелика програма, в якій можна створити зручну базу даних власних кулінарних рецептів. Програма - умовно-безкоштовна. Сайт розробника – www.valksoft.narod.ru.

У цьому кулінарному довіднику є одна важлива перевага. При додаванні нового рецепта передбачена можливість сформувати список інгредієнтів із зазначенням кількості і одиниці вимірювання. Серед недоліків – відсутня можливість додати фотоілюстрації.

Інтелектуальна пошукова система дозволяє здійснювати пошук за назвою, продуктам, підбирати рецепт по конкретному товарно-продовольчому ряду.

За замовчуванням в базі даних міститься 501 рецепт, але з сайту можна скачати великий набір тематичних кулінарних колекцій та імпортувати їх в базу даних.

Програма має чудову навігацію, продумані функції і опції (рис. 1.3.).

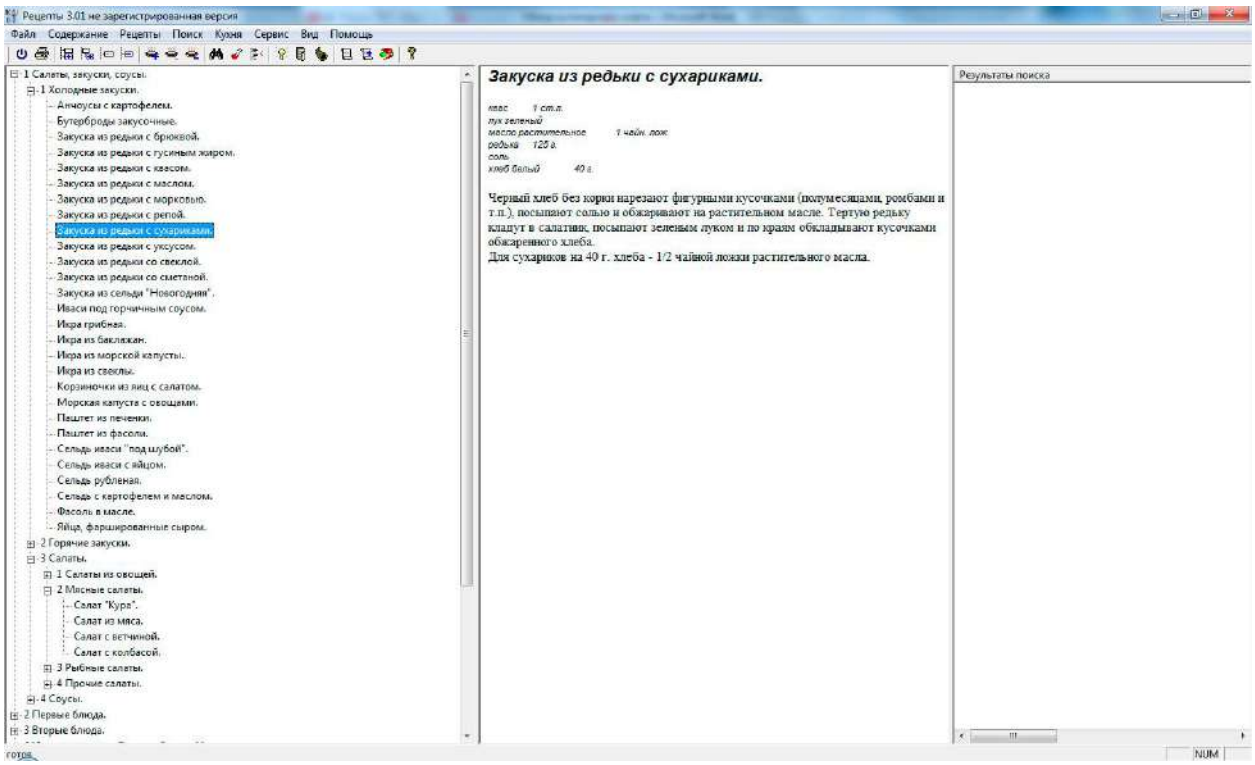

Рис. 1.3. Головне меню програми «Рецепти 3.0.1.»

Функціональні можливості програми:

- Додавання нових розділів, редагування і видалення розділів у змісті.
- Додавання нових рецептів, редагування і видалення рецептів.
- Пошук рецептів за назвою.
- Пошук рецептів за обраним продукту.
- Пошук рецептів за наявними в наявності продуктам.
- Друк рецептів на принтер.
- Імпорт-експорт розділів з рецептами.

4. Книга кулінарних рецептів Recepies 1.0.

Ця програма від того ж розробника Valksoft. Вона є логічним продовженням програми «Рецепти 3.0.1». У ній передбачена велика кількість сучасних переваг і поліпшень. Покращена навігація, з’явилися нові функції, рецепт можна формувати покроково і надавати на кожному кроці ілюстрації (рис. 1.4.). Для списку інгредієнтів передбачено окреме вікно.

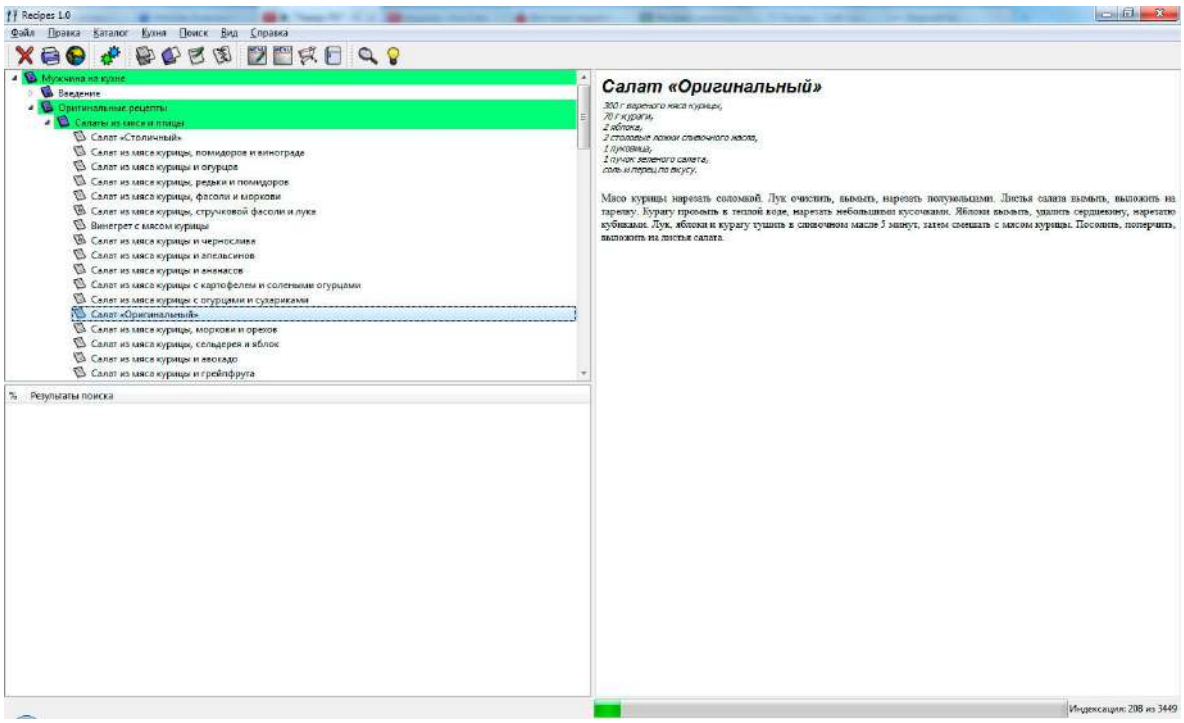

Рис. 1.4. Головне меню програми «Книга кулінарних рецептів Recepies 1.0»

Загалом, в цій програмі є все, що необхідно для складання зручного каталогу кулінарних рецептів. При необхідності в базу даних програми можна імпортувати кулінарні колекції з сайту розробника - www.kobelkov.ru. І головна перевага програми – вона безкоштовна.

Функціональні можливості програми:

- Редагування рецептів, видалення і введення нових рецептів.
- Вставка картинок в рецепти.
- Пошук рецептів за ключовими словами.
- Пошук рецептів за наявними продуктами.
- Отримання списку продуктів для складеного меню.
- Виведення на друк вибраного рецепту, або списку продуктів для складеного меню.
- Імпорт-експорт для обміну рецептами між програмами.

5. My Cookery Book 5.02.

Програма має приємний інтерфейс, навігацію, оптимальну кількість функцій, можливість додавати фотографії, голосові підказки, інтелектуальний пошук. При всьому цьому програма безкоштовна (рис. 1.5.).

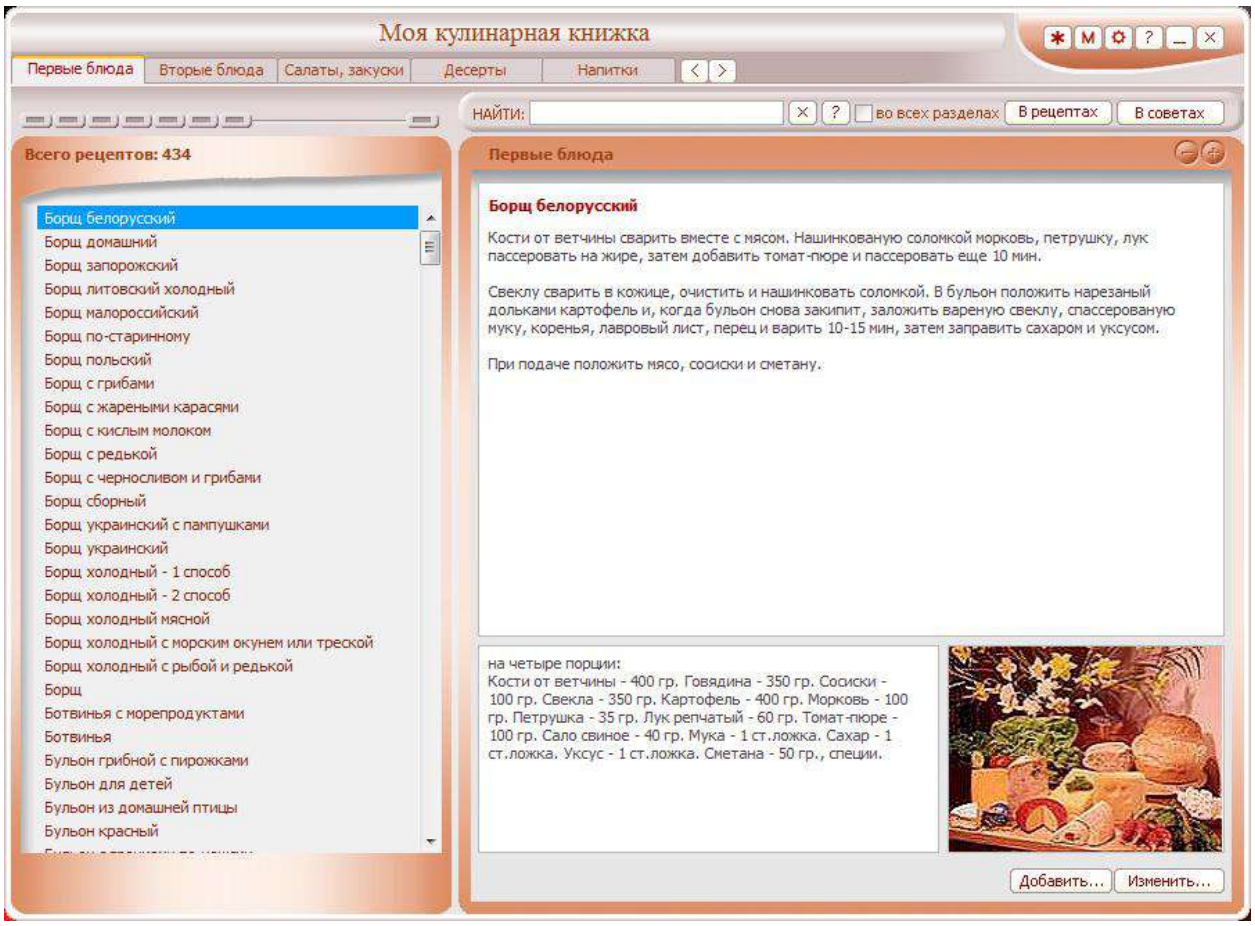

Рис. 1.5. Головне меню програми «My Cookery Book 5.02»

6. Нотатки кулінара 1.11.

Програма з базою кулінарних рецептів і можливістю додавати свої власні. Сайт програми - www.cn.graydon.ru. На сайті можна скачати колекції кулінарних збірок, після чого імпортувати їх в базу даних програми.

Інтерфейс реалізований гарно, кількість опцій при додаванні нового рецепта мінімізована і добре впорядкована. Окреме поле передбачено для інгредієнтів, є можливість додати одне фото для кожного рецепту. Також розробник передбачив окреме поле для вказівки джерела рецепта, дозволив оцінити рецепт за п’ятибальною шкалою і позначити кількість порцій. Категорія вибирається безпосередньо в картці рецепта (рис. 1.6.).

Функціональні можливості програми:

- Перегляд списку рецептів.
- Додавання нових рецептів.
- Внесення змін до існуючих рецептів.

- Пошук по тексту рецептів.
- Друк обраного рецепта.

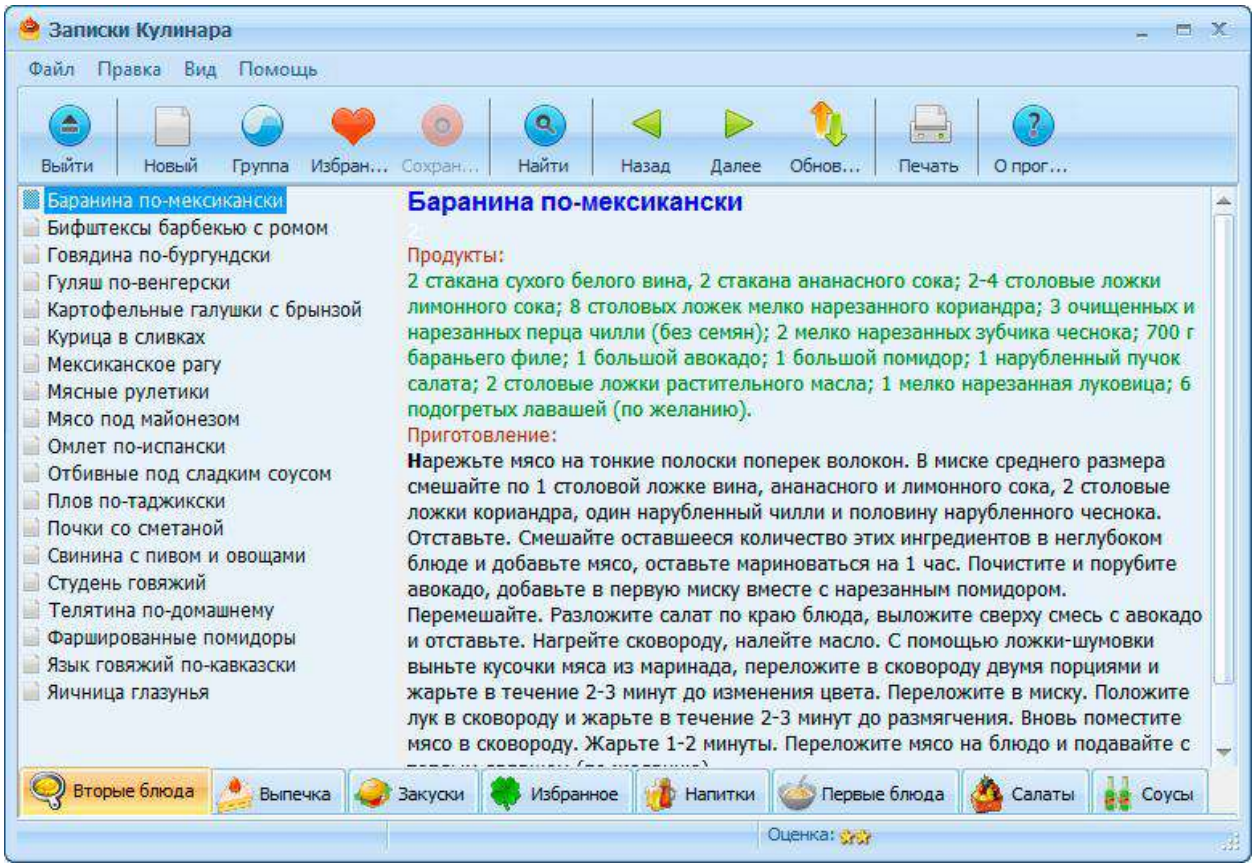

Рис. 1.6. Головне меню програми «Нотатки кулінара 1.11»

7. Кулінар 9.8.

Якщо попереднє програмне забезпечення можна назвати аматорським, домашнім, то програма «Кулінар 9.8», безумовно, професійний продукт, створений для автоматизації виробничих процесів ресторанів, кафе та інших буфетів.

На сайті розробника - www.softbis.narod.ru - стверджується, що програма умовно-безкоштовна.

Варто зазначити, що програма не проста, не адаптована на 100% до некваліфікованого користувача. Для роботи з нею необхідно мати навички щодо підтримки і ведення баз даних. При цьому на виробництві вона, однозначно, допоможе оптимізувати ряд бізнес-процесів (рис. 1.7.).

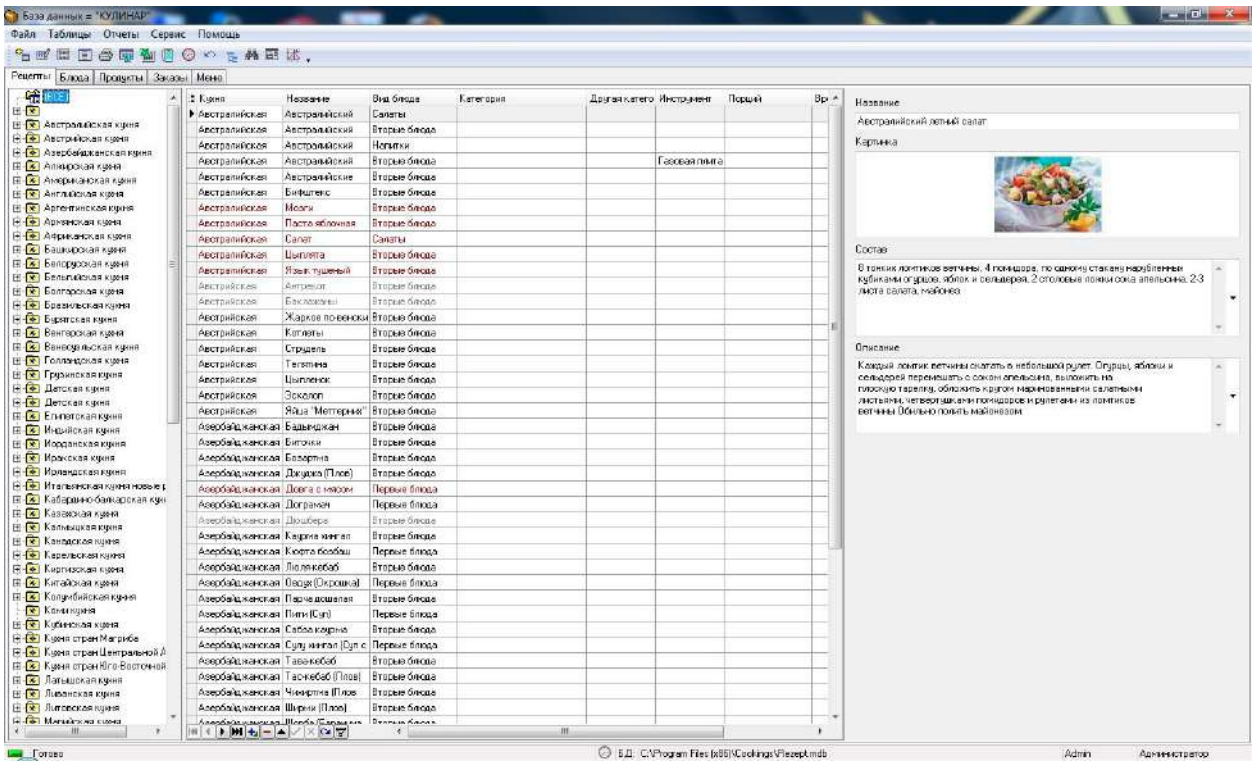

Рис. 1.7. Головне меню програми «Кулінар 9.8»

В базі даних міститься більше 12 000 рецептів. З її допомогою можна в автоматичному режимі складати калькуляційні карти, меню, відображати рух продуктів на складі. Також передбачено сортування по декількох полях, фільтри, угрупування, пошук. Будь-які таблиці бази даних можна експортувати в поширені формати даних або навпаки імпортувати дані в бази даних з різних джерел.

Функціональні можливості програми:

- Можна створювати, змінювати, видаляти записи, поля, таблиці.
- Можна імпортувати дані в будь-яку таблицю бази даних з текстових файлів.
- Можна сортувати таблиці по будь-якому полю.
- Можна фільтрувати таблицю по будь-якому полю.
- Можна додавати записи як «Вибрані», тоді вони будуть відображатися помаранчевим кольором. Колір задається у властивостях таблиці.

- Можна додавати записи як «Мертві» («нецікаво»), тоді вони будуть відображатися сірим (або іншим) кольором.
- Можна налаштовувати правила виділення кольором. Ви самі визначаєте, які рядки, яким кольором і за яких умов виділяти.
- Можна будувати дерево по будь-яких полях з довільною кількістю рівнів для ієрархічного відображення даних будь-якої таблиці.
- Можна змінювати дані в будь-якому полі (крім ID і обчислюваних полів) прямо в таблиці або в окремій формі (вибирається в налаштуваннях).
- Можна створювати нові збережені поля для таблиць наступних типів: текстове, числове, Так / Ні, Дата і час.
- Можна створювати обчислювані поля для таблиць, наприклад можна створити поле з формулою «[Поле 1] / [Поле 2]».
- Можна створювати обчислювані поля, значення яких будуть братися з інших таблиць. Наприклад, можна вивести ім’я боржника з таблиці «Боржники».
- Можна створювати нові таблиці з абсолютно такими ж можливостями по діях з ними, як і у будь-який іншій таблиці.
- Можна прив’язувати спадаючі списки полів до інших таблиць для легкого вибору значень з них при редагуванні в таблиці або для вибору з інших форм при редагуванні у формі.
- Можна ставити будь-яку кількість підлеглих таблиць для будь-якої таблиці, для чого необхідно задати прив’язку по полях у властивостях таблиці.
- Можна змінювати порядок проходження полів у будь-якій таблиці.
- Можна перейменовувати поля таблиць і назви самих таблиць у відповідності зі специфікою.
- Можна друкувати поточне подання будь-якої таблиці з урахуванням видимості полів, їх ширини і порядку.

- Можна експортувати дані будь-якої таблиці в MS Excel, MS Word з урахуванням поточного подання таблиці.
- Можна працювати з декількома файлами баз даних, створювати нові бази даних, також можна їх відкривати за допомогою MS Access.

Висновки до розділу 1

Підводячи висновок першого розділу дипломної роботи, слід сказати, що розробка програмного забезпечення кулінарних рецептів є актуальною задачею. При даній розробці необхідно правильно продумати та розробити структурну схему бази даних та створити відповідне програмне забезпечення. Враховуючі розглянуті аналоги програм кулінарних довідників потрібно врахувати всі їх недоліки та розробити повноцінне програмне забезпечення.

РОЗДІЛ 2
РОЗРОБКА АЛГОРИТМУ РОЗВ'ЯЗАННІ ЗАДАЧІ

2.1. Розробка алгоритму вирішення задачі «Cooking»

При розробці програмного забезпечення необхідно створити функціональну схему задачі. У нашому випадку програма повинна виконувати наступні функціональні задачі:

1. Ведення та редагування даних довідників;
2. Ведення та редагування даних страв та продуктів;
3. Формування та виведення звітної інформації.

На рис. 2.1. наведемо функціональну схему даної задачі.

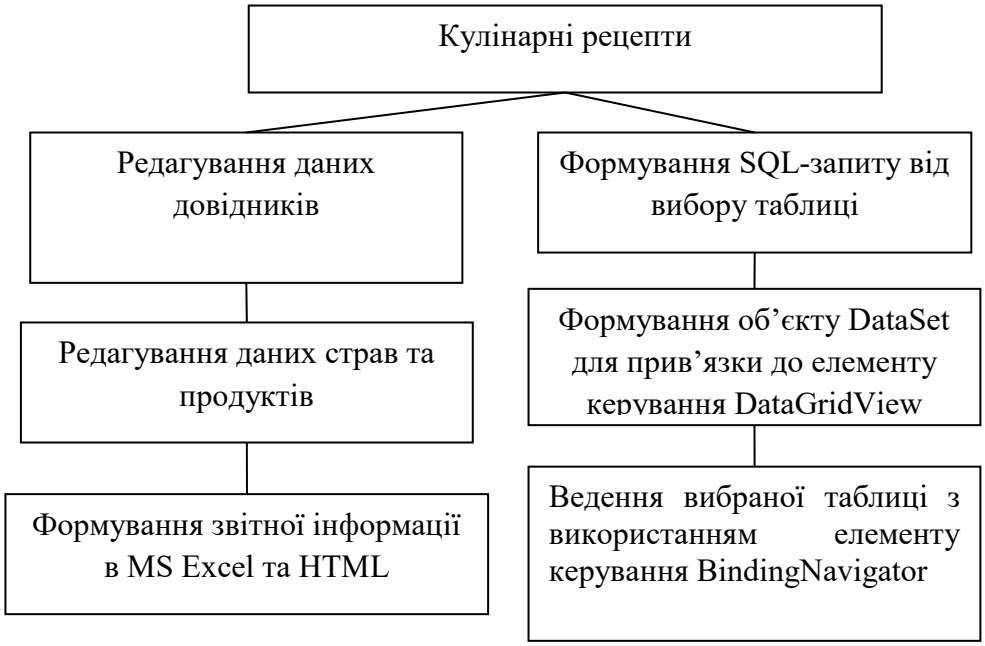

Рис. 2.1. Функціональна схема задачі «Кулінарні рецепти»

Нижче наведемо блок-схему роботи програми (рис. 2.2.).

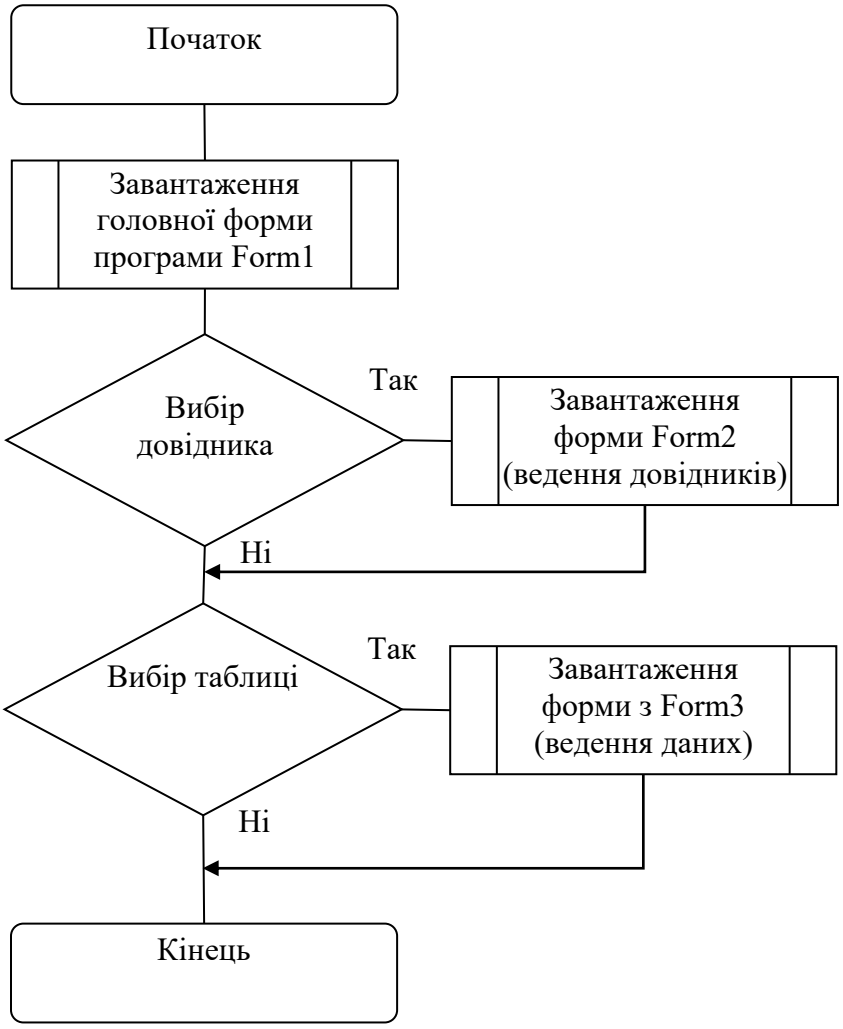

Рис. 2.2. Блок-схема задачі «Кулінарні рецепти»

2.2. Розробка алгоритму вирішення задачі «Cooking_3D»

Для реалізації задачі необхідно розробити алгоритм. Робота з програмою «Cooking_3D» здійснюється на алгоритму, який зображено на рис. 2.3.

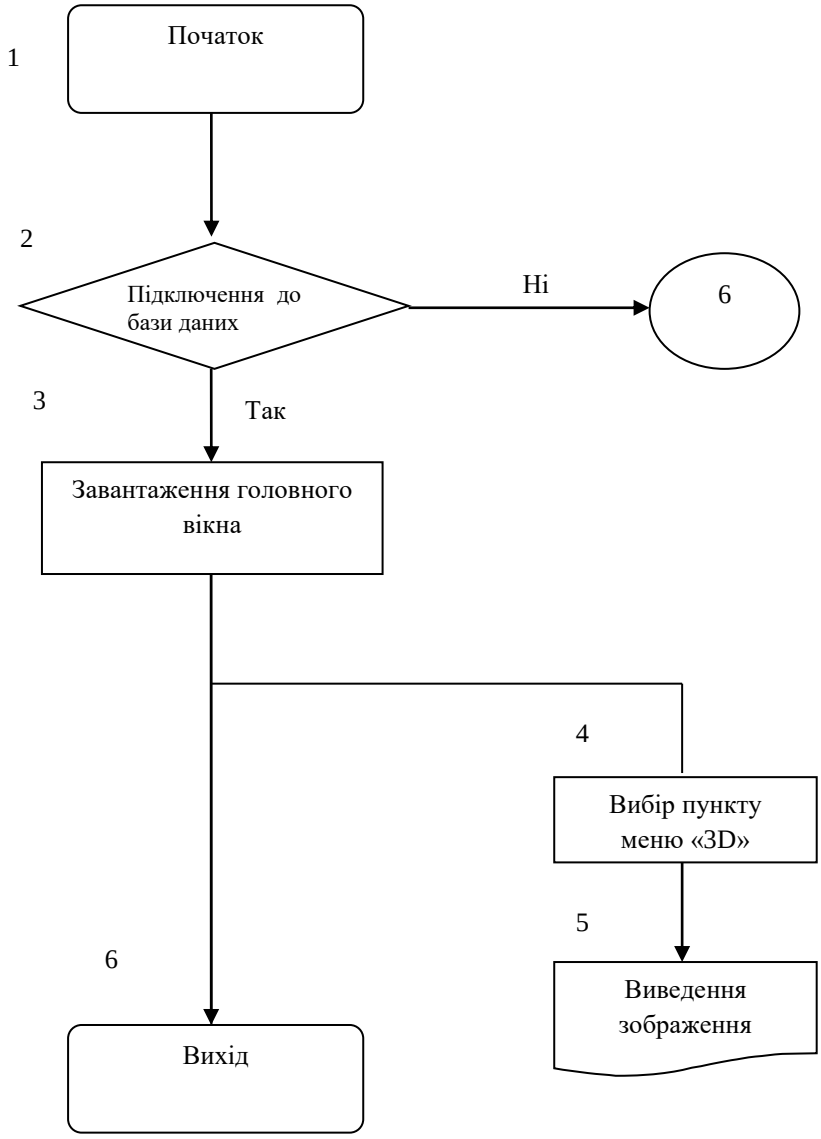

Рис. 2.3. Блок-схема алгоритму роботи програми «Cooking_3D»

Висновки до розділу 2

Отже, у цьому розділі було успішно розроблено алгоритми вирішення задач «Cooking», «Cooking_3D» шляхом побудови блок-схем алгоритмів цих задач. При створені алгоритми для вирішення цих задач було використано повний набір вхідних та вихідних даних задач.

РОЗДІЛ 3
ОРГАНІЗАЦІЯ ІНФОРМАЦІЙНОГО ЗАБЕЗПЕЧЕННЯ

3.1. Структура бази даних

Розглянемо структурну модель бази даних.

У якості середовища баз даних вибрано СУБД ACCESS. База даних має назву «Кулинария.mdb» та зберігається у поточному каталозі. БД складається з семи таблиць: CategoriesDish – довідник категорій страв; CategoriesProduct – довідник категорій продуктів; Country – довідник по країнам; Dishes – інформація по стравам; Products – інформація по продуктам; Recepies – інформація по рецептам для приготування страв з продуктів; Terms – умови проведення заходів (Весілля, День народження тощо).

У табл. 3.1. наведемо структуру таблиці CategoriesDish (ID – ключове поле).

Таблиця 3.1

Структура таблиці CategoriesDish

№ п/п	**Назва**	**Тип поля**	**Значення**
1	**ID**	Лічильник	Забезпечення нормалізації та реляції даних
2	Name	Текстове	Назва категорії страв

У табл. 3.2. наведемо структуру таблиці CategoriesProduct (ID – ключове поле).

Таблиця 3.2

Структура таблиці CategoriesProduct

№ п/п	**Назва**	**Тип поля**	**Значення**
1	**ID**	Лічильник	Забезпечення нормалізації та реляції даних
2	Name	Текстове	Назва категорії продуктів

У табл. 3.3. наведемо структуру таблиці Country (ID – ключове поле).

Таблиця 3.3

Структура таблиці Country

№ п/п	**Назва**	**Тип поля**	**Значення**
1	**ID**	Лічильник	Забезпечення нормалізації та реляції даних
2	Name	Текстове	Назва країни

У табл. 3.4. наведемо структуру таблиці Dishes (ID – ключове поле).

Таблиця 3.4

Структура таблиці Dishes

№ п/п	**Назва**	**Тип поля**	**Значення**
1	**ID**	Лічильник	Забезпечення нормалізації та реляції даних
2	ID_cat	Довге ціле	Зв’язка з полем ID таблиці CategoriesDish
3	Name	Текстове	Назва страви
4	Kol	Довге ціле	Кількість страв
5	Cena	Подвійне з плаваючою точкою	Ціна однієї страви

У табл. 3.5. наведемо структуру таблиці Products (ID – ключове поле).

Таблиця 3.5

Структура таблиці Products

№ п/п	**Назва**	**Тип поля**	**Значення**
1	**ID**	Лічильник	Забезпечення нормалізації та реляції даних
2	ID_cat	Довге ціле	Зв’язка з полем ID таблиці CategoriesProduct
3	Name	Текстове	Назва продукту
4	Kol	Довге ціле	Фасовка, г

Продовження таблиці 3.5

№ п/п	Назва	Тип поля	Значення
5	Calorie	Довге ціле	Калорійність
6	Belki	Довге ціле	Білки
7	Zhiry	Довге ціле	Жири
8	Uglev	Довге ціле	Вуглеводи

У табл. 3.6. наведемо структуру таблиці Recepies (ID – ключове поле).

Таблиця 3.6

Структура таблиці Recepies

№ п/п	Назва	Тип поля	Значення
1	**ID**	Лічильник	Забезпечення нормалізації та реляції даних
2	ID_Recepies	Довге ціле	Номер рецепту
3	ID_country	Довге ціле	Зв’язка з полем ID таблиці Country
4	ID_Dish	Довге ціле	Зв’язка з полем ID таблиці Dishes
5	ID_Product	Довге ціле	Зв’язка з полем ID таблиці Products
6	Kol	Довге ціле	Кількість продуктів для певної страви

У табл. 3.7. наведемо структуру таблиці Terms (ID – ключове поле).

Таблиця 3.7

Структура таблиці Terms

№ п/п	Назва	Тип поля	Значення
1	**ID**	Лічильник	Забезпечення нормалізації та реляції даних
2	Name	Текстове	Назва умови проведення заходів

Дані таблиці пов’язані між собою відношеннями один до багатьох.

На рис. 3.1. наведемо схему даних бази даних «Кулинария.mdb». З даної схеми видно реляційні зв’язки типу один-до-багатьох. Так, таблиця CategoriesDish пов’язана з таблицею Dishes відношеннями один-до багатьох, тобто у кожної категорії страви може бути багато страв. Таблиця CategoriesProduct пов’язана з таблицею Products відношеннями один-до багатьох, тобто у кожної категорії продуктів має місце багато продуктів.

Безпосередньо таблиця, яка характеризує список рецептів «Recepies» має зв’язки з трьома таблицями, які виконують роль довідників, а саме: Country (список країн), Dishes (список страв), Products (список продуктів).

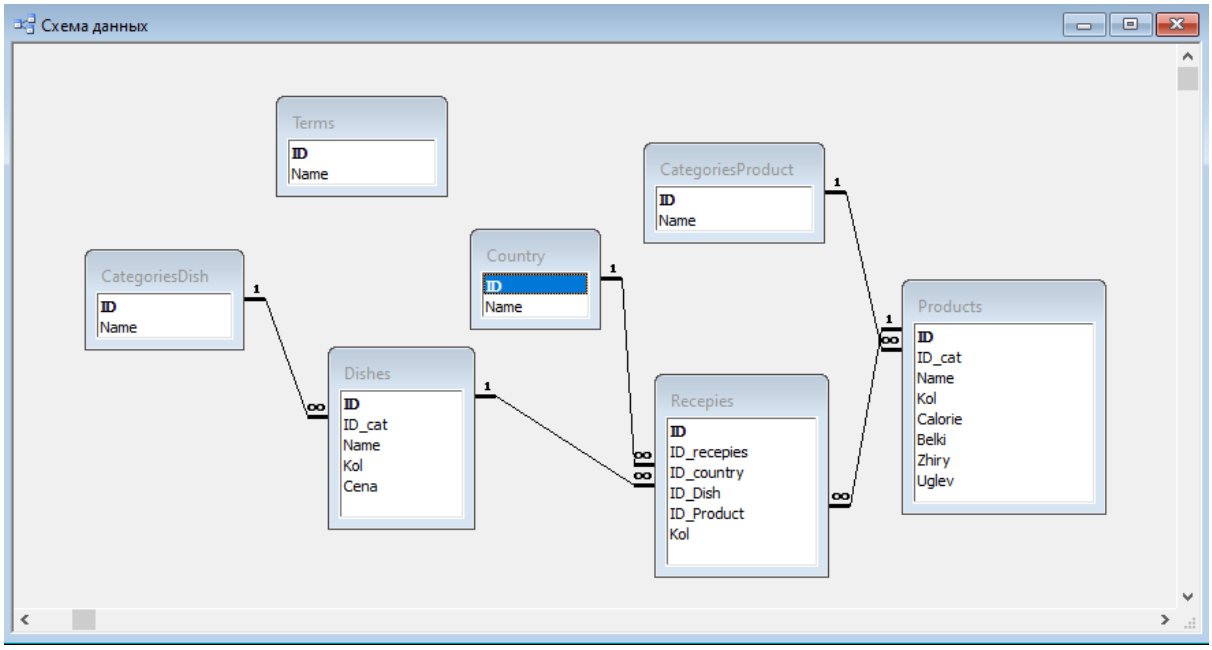

Рис. 3.1. Схема даних бази даних «Кулинария.mdb»

Дана структурна модель бази даних дозволяє повністю автоматизувати процес реалізації програмного забезпечення «Кулінарні рецепти».

3.2. Розробка діаграм-класів програмного комплексу

Програмний комплекс складається з семи візуальних класів *Form1-Form7* та класів *Name_List* та *Name_Product*, які призначені для прив’язки до

елементів керування, таких як *ListBox, ComboBox, DatagridView*. Дані діаграми класів робляться в пакеті Visual Studio за допомогою **View Class Diagram** (альтернатива UML-діаграм) [додаток А].

На рис. 3.2. наведена діаграма класів *Form1-Form3* [додаток Б, В, Г].

Form1 – головна форма програми, в якій зберігається об'єкт підключення до бази даних *conn* типу *OleDBConnection*, інформація з налаштування кольору та шрифту звітної інформації (*cvet_hapka, cvet_text, cvet_zag, font_text*). Сірим кольором шрифту виділені елементи керування. До даної форми належать тільки опції меню. Рожевим кольором шрифту виділенні повідомлення головної форми, а також власні методи: *Export_HTML* – універсальна функція експорту в HTML-формат; *ExportXLS* – експорт до формату *MS Excel*. Проектування форми 1 наведено на рис. 3.3.

Form2 – форма ведення довідників таблиць бази даних: *категория блюд*, *категория продуктов*, *страны рецептов*, *условия мероприятий*. Для роботи з довідниками бази даних використовуються наступні елементи керування: *bindingSource* – універсальний зв'язувач набору даних *DataTable* з елементами керування. У нашому випадку це наступні елементи: рядок навігації *BindingNavigator* та елемент керування *DatagridView*. При проектуванні форми 2 рядок навігації було розширено наступними можливостями, а саме: додавання кнопок експорту даних в HTML та XLS. Проектування форми 2 наведено на рис. 3.4. Також у формі 2 присутні об'єкти *DataAdapter* (виконує обмін даними між таблицею *DataTable* та таблицею бази даних), *DataSet* (вміщує колекцію таблиць *DataTable*), *OleDBCommandBuilder* (автоматично формує три запити до бази даних: додавання – INSERT INTO, видалення – DELETE, та редагування інформації з таблиці бази даних – UPDATE).

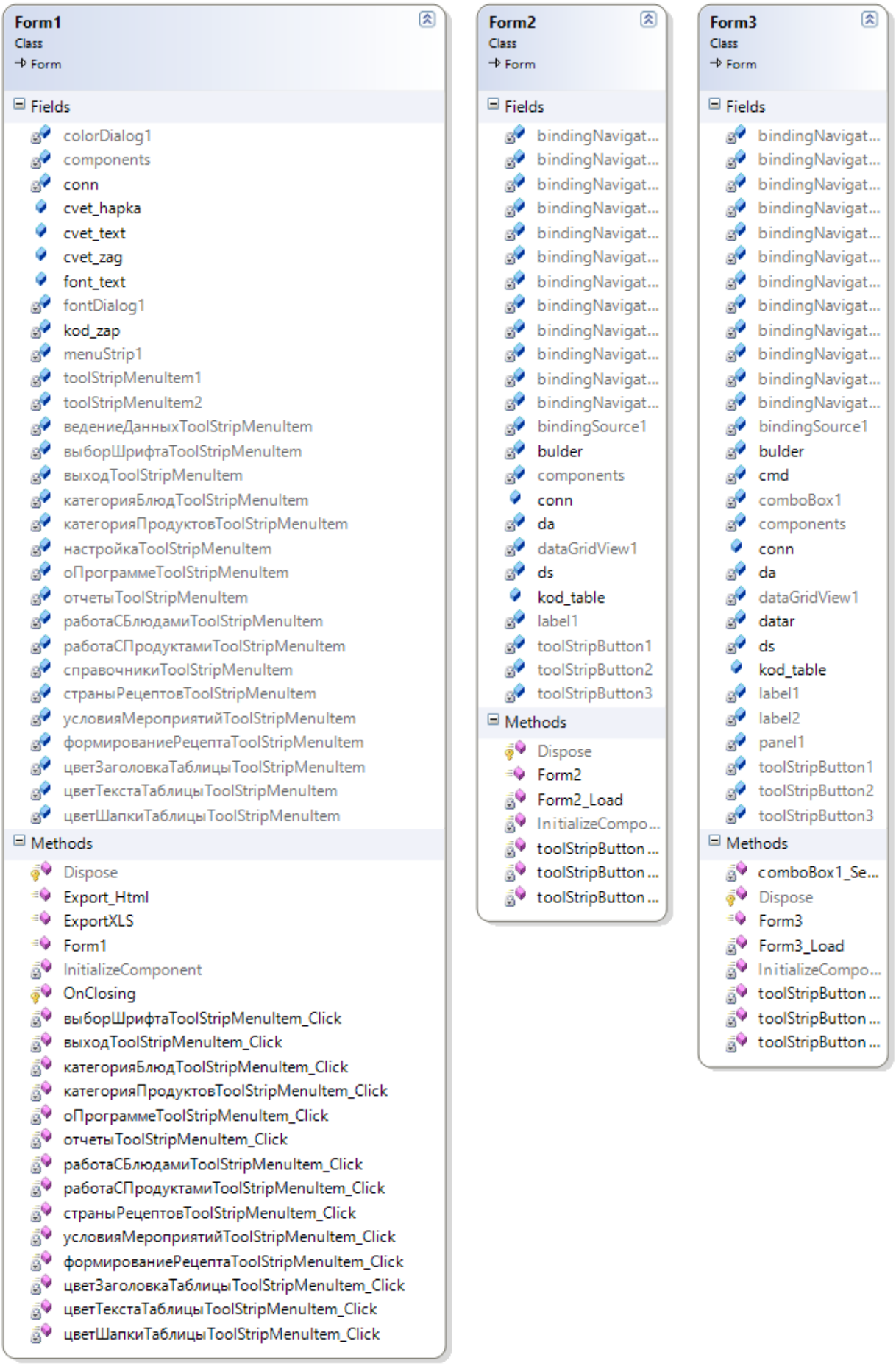

Рис. 3.2. Діаграма класів Form1-Form3.

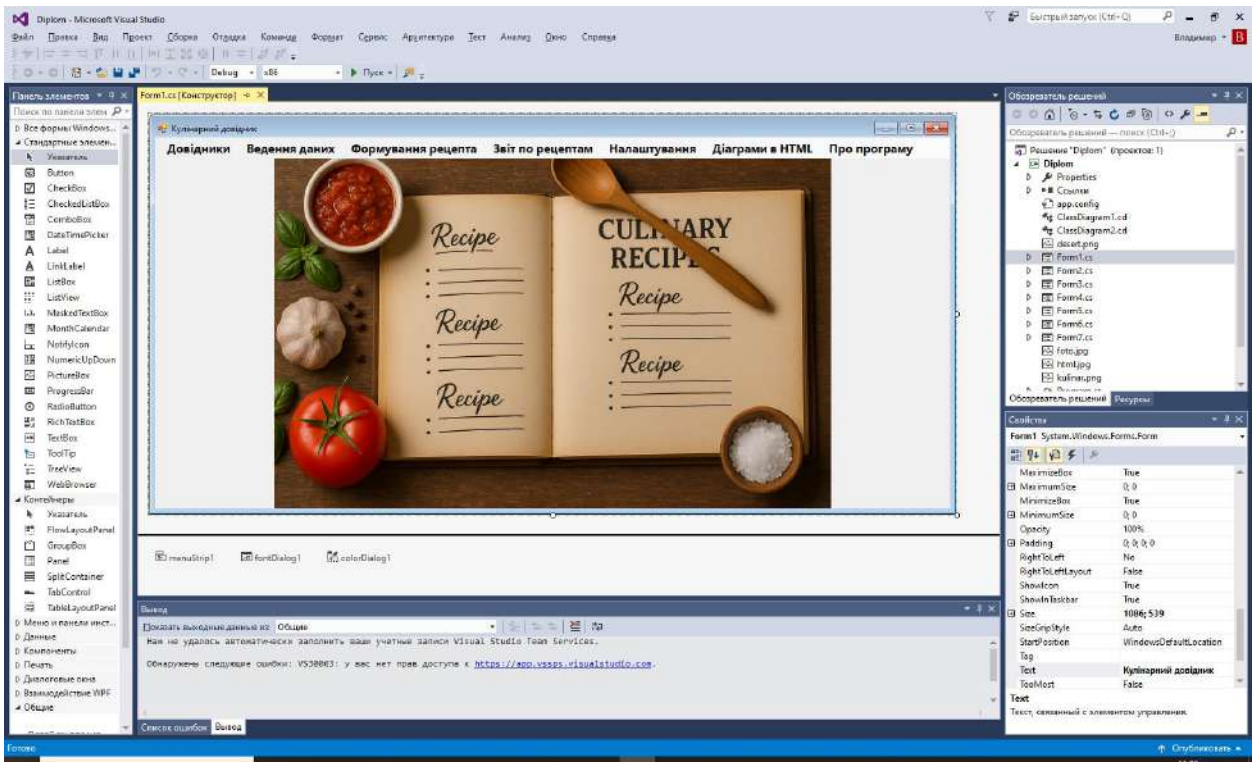

Рис. 3.3. Проектування інтерфейсу класу Form1

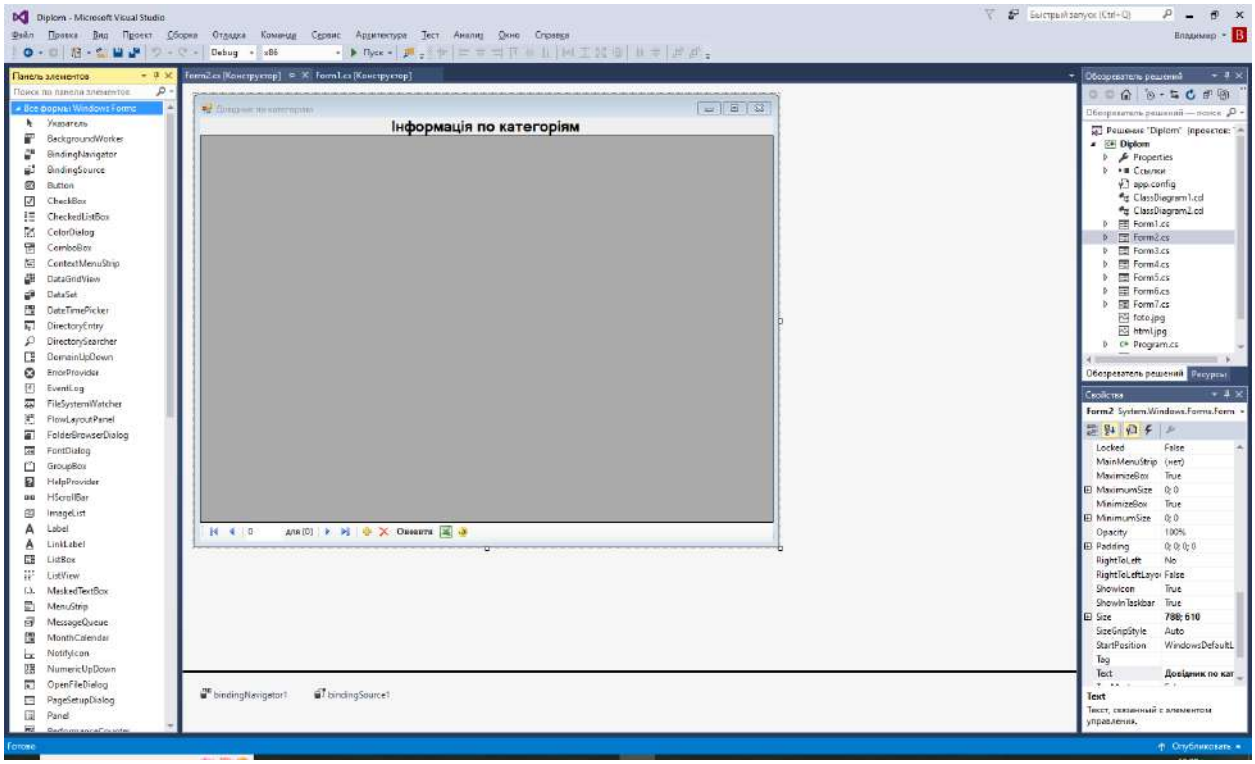

Рис. 3.4. Проектування інтерфейсу класу Form2

Form3 – форма ведення даних за реляційним зв’язком один-до-багатьох: *страви*, *продукти*. У даній формі з’являється елемент керування ComboBox у якому вибирається інформація з довідника по відповідній

категорії. Потім, по даній категорії, за допомогою запиту на вибірку SELECT заповнюється елемент керування *DataGridView*. *ComboBox* заповнюється за допомогою об’єкту *DataReader*, який формується за допомогою об’єкту *OleDBCommand*. Проектування форми 3 наведено на рис. 3.5.

На рис. 3.6. наведена діаграма класів *Form4-Form7*.

Form4 – форма експорту даних в формат HTML. В формі розміщено елемент керування WebBrowser для відображення інформації у форматі HTML. Проектування форми 4 наведено на рис. 3.7.

Form5 – форма, призначена для формування рецептів страв та запису їх до бази даних. У даній формі формується контейнер типу *List*, який має назву *list_product*. Даний контейнер динамічно прив’язується до елементу керування *DataGridView* за допомогою *BindinSource*. Інші дані вибираються з елементів керування *ComboBox*. Проектування форми 5 наведено на рис. 3.8. При натисканні кнопки «Формування рецепта», відповідний рецепт за допомогою запиту INSERT INTO записується до бази даних.

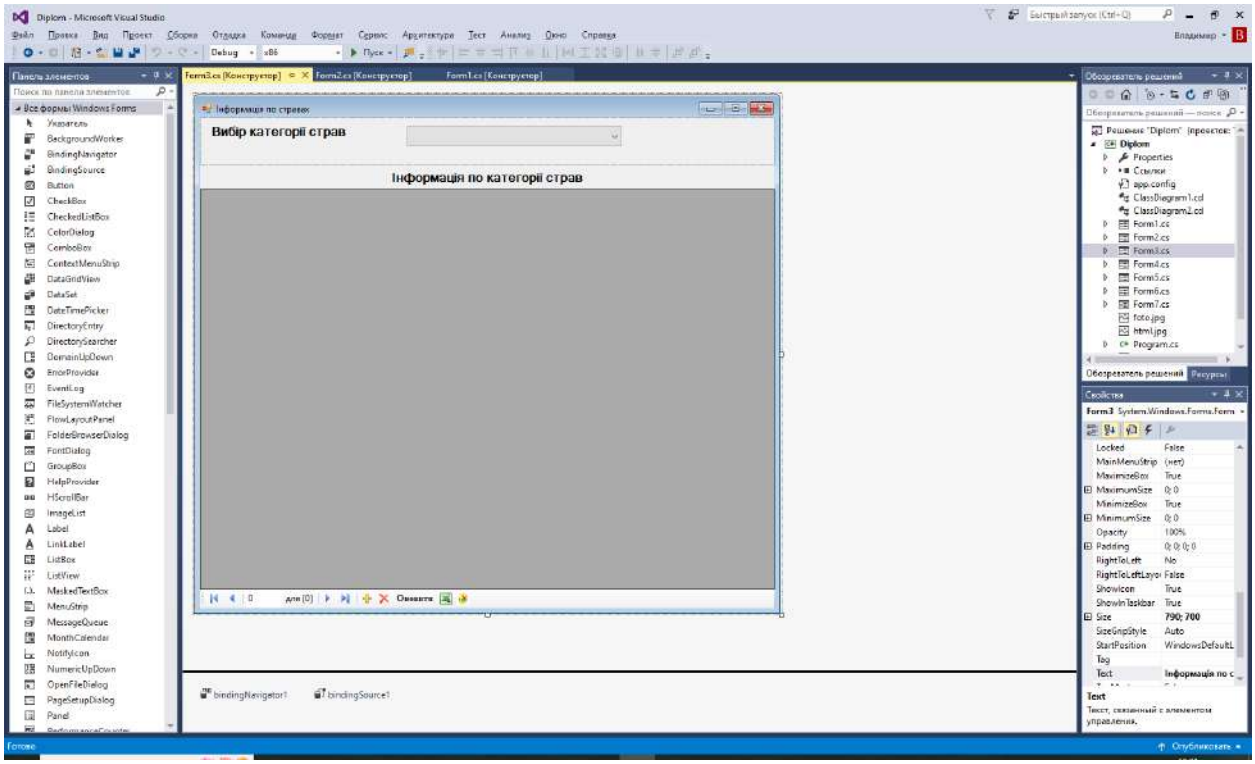

Рис. 3.5. Проектування інтерфейсу класу Form3

Form6 – форма звітності по обраним рецептам у MS Excel. У даній формі з елементу керування *ComboBox* обирається номер рецепту. Далі заповнюються елементи керування *TextBox* (Країна, Страва) та *DatagridView* (вибрані продукти для приготування страви за рецептом). Проектування форми 6 наведено на рис. 3.9. При натисканні кнопки «Експорт в XLS», формується звіт по вибраному рецепту у формат XLS.

Form7 – інформація про розробника програмного забезпечення. Проектування форми 7 наведено на рис. 3.10.

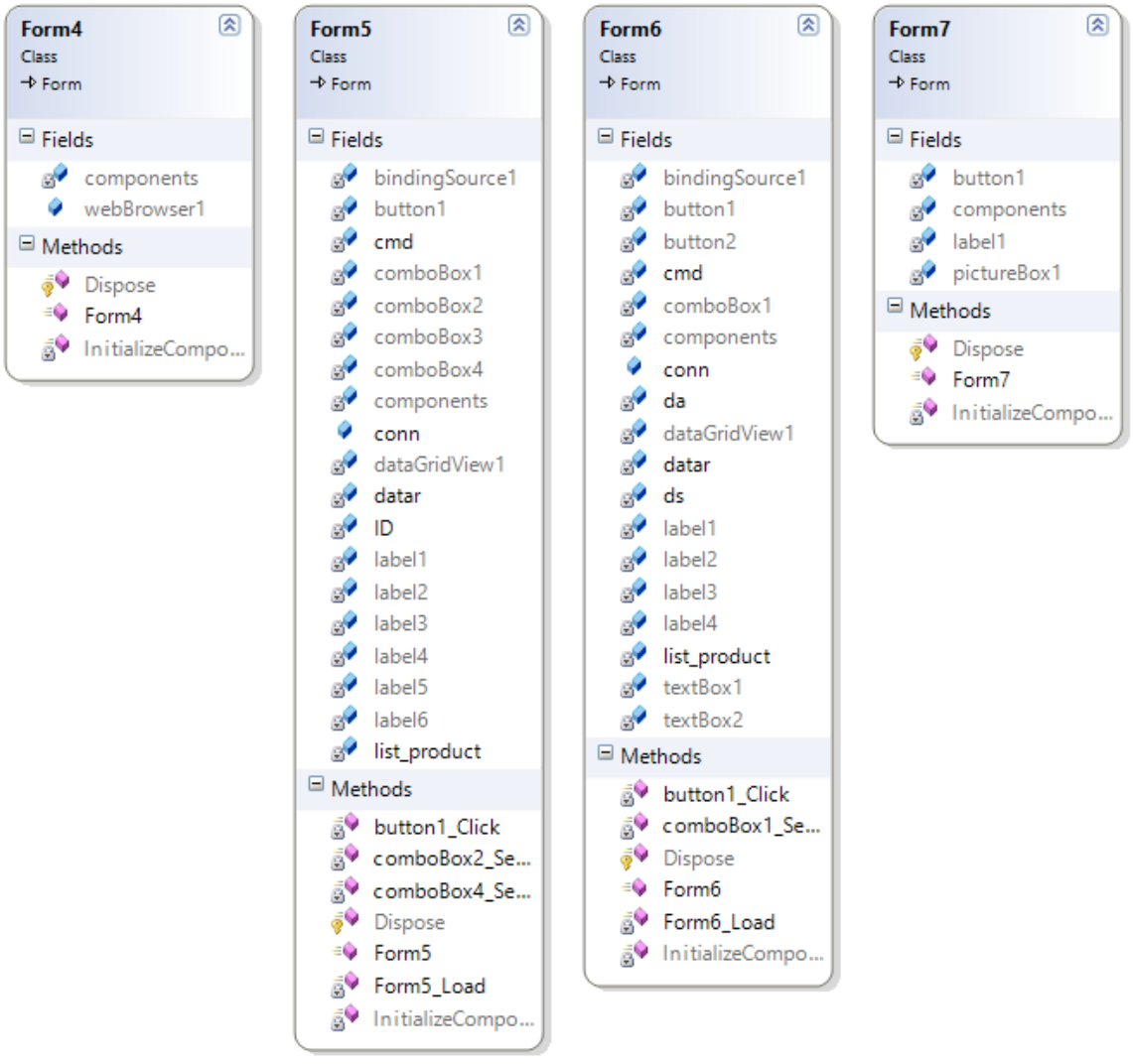

Рис. 3.6. Діаграма класів Form4-Form7

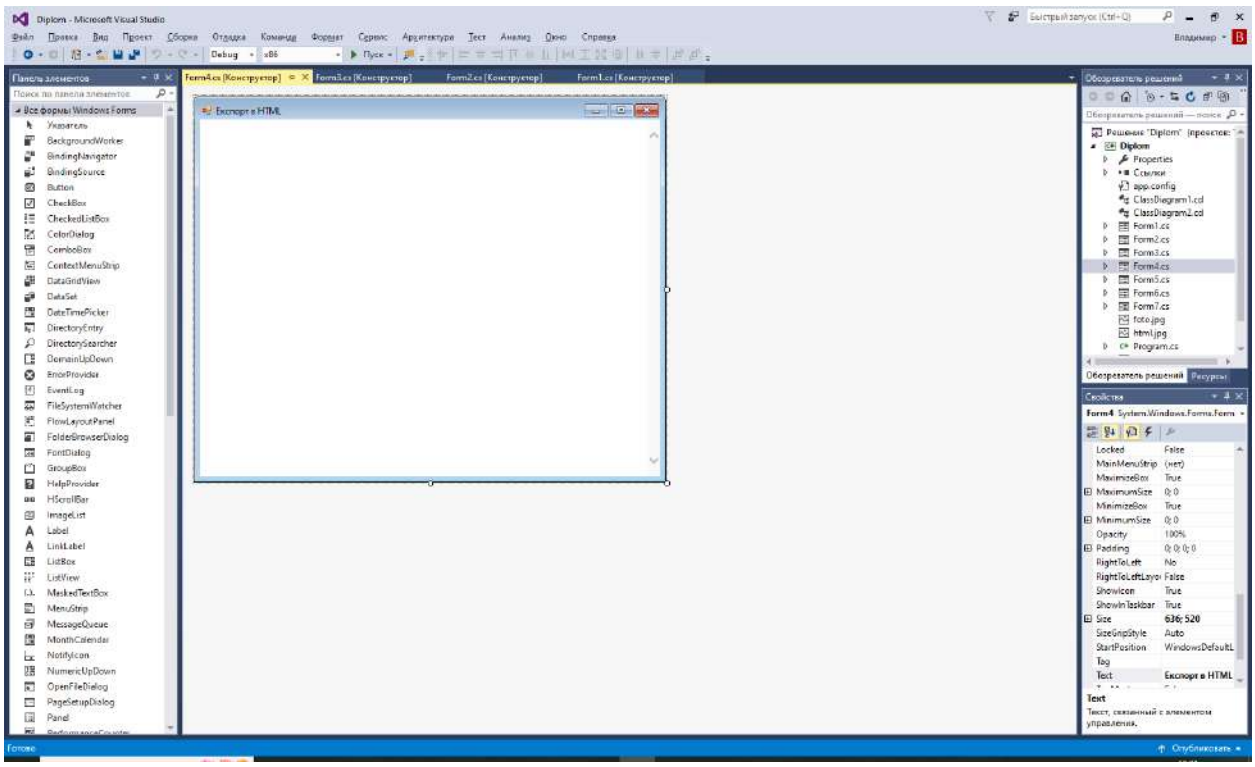

Рис. 3.7. Проектування інтерфейсу класу Form4

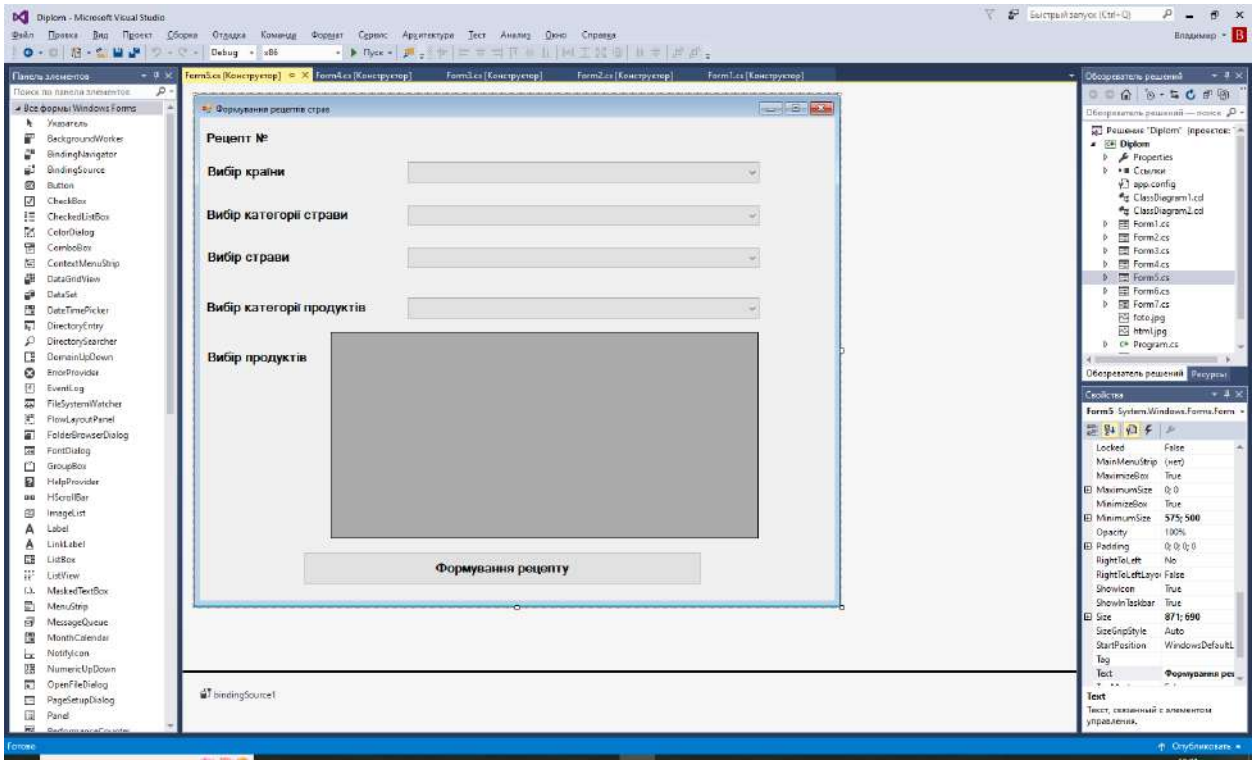

Рис. 3.8. Проектування інтерфейсу класу Form5

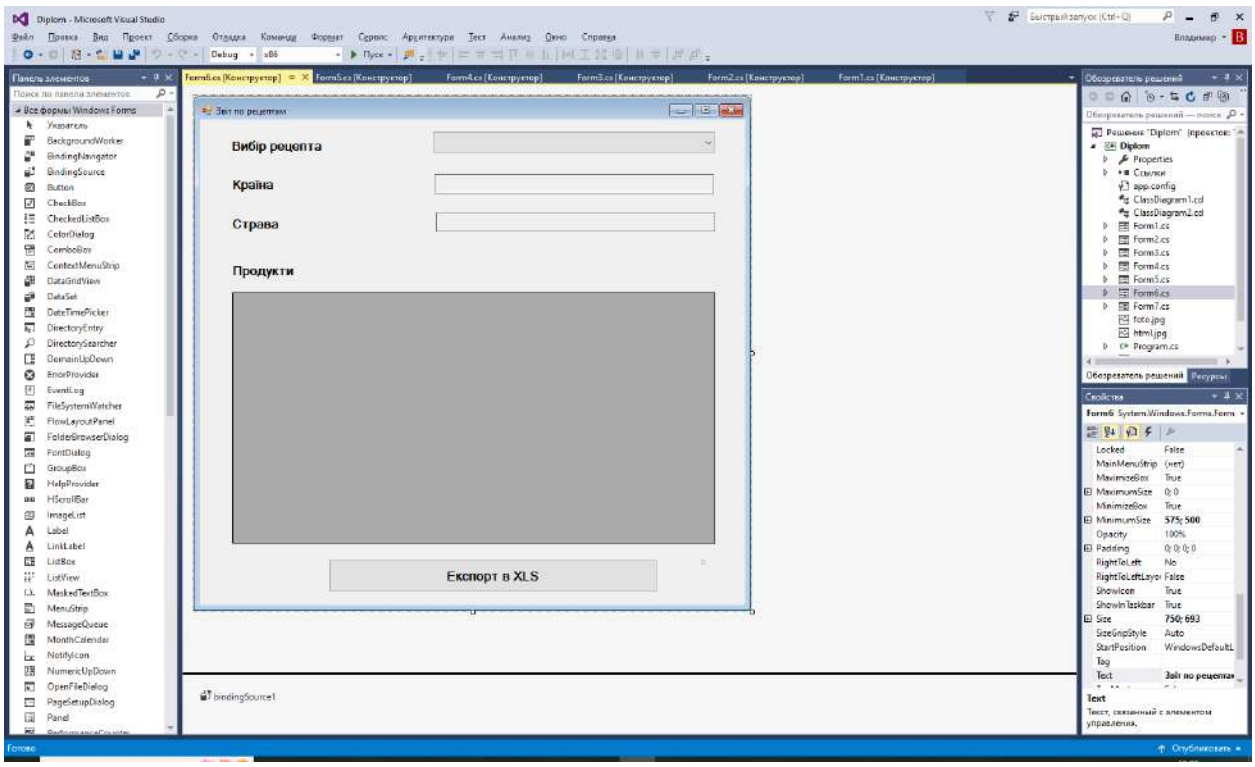

Рис. 3.9. Проектування інтерфейсу класу Form6

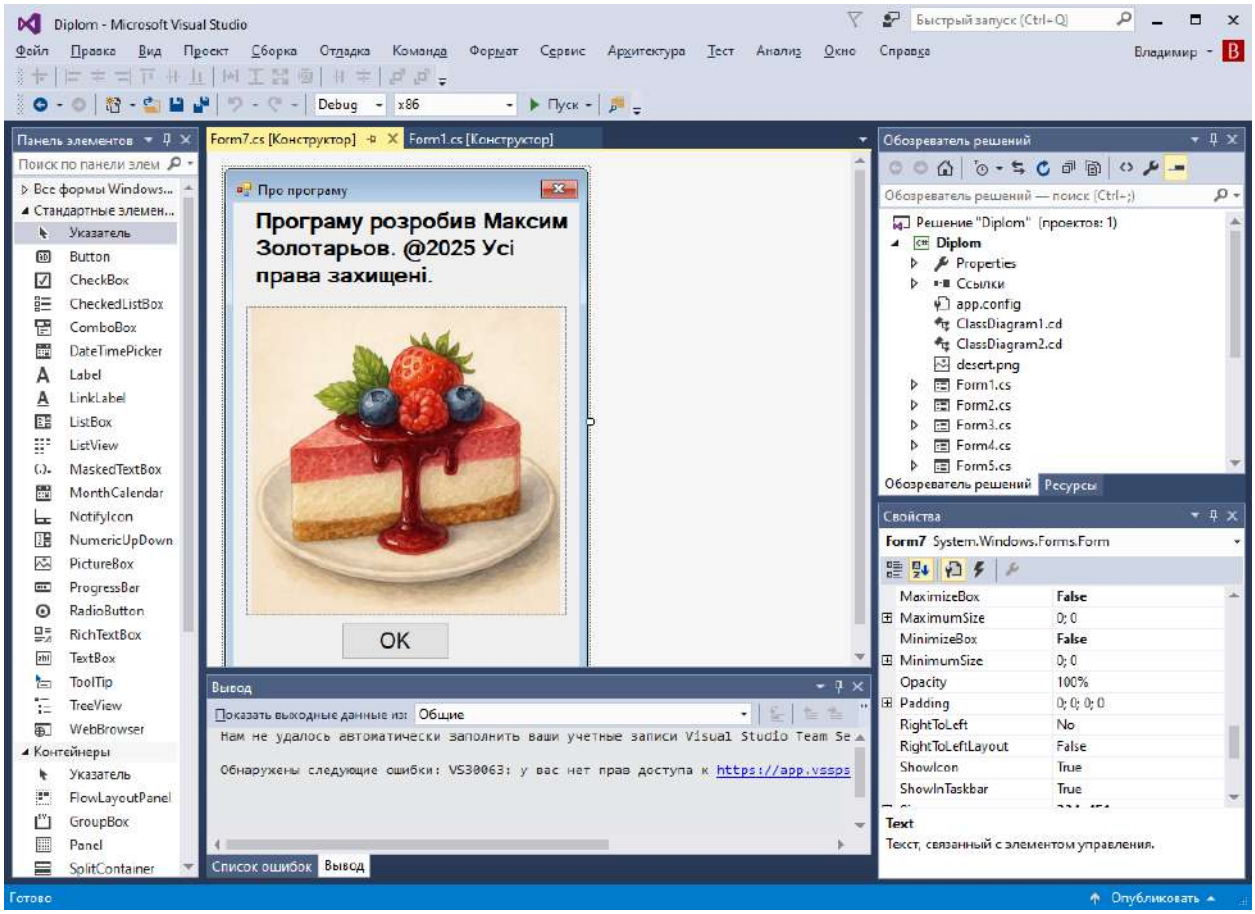

Рис. 3.10. Проектування інтерфейсу класу Form7

На рис. 3.11. наведена діаграма класів *Name_List* та *Name_Product*.

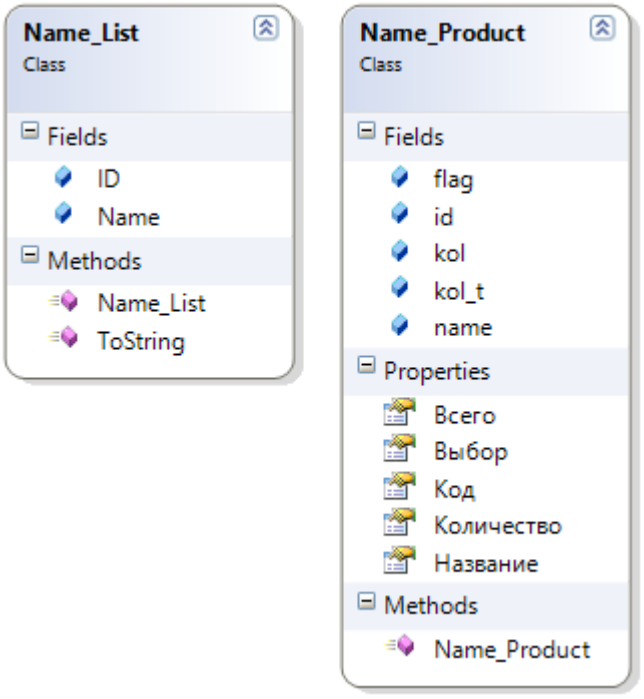

Рис. 3.11. Діаграма класів Name_List та Name_Prroduct.

Дані класи призначені для універсальної прив'язки до елементів керування: *Name_List* – для прив'язки довідників до елементів керування; *Name_Product* – для прив'язки контейнеру *list_product* до елементу керування *DataGridView* при формуванні рецептів.

Висновки до розділу 3

Отже, у цьому розділі було детально розглянуто структуру бази даних для розробки програмного забезпечення кулінарних рецептів. Крім того розглянуто проектування програмного забезпечення, а також наведені діаграми класів, як по функціональній частині так і по візуальних формах.

РОЗДІЛ 4
РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ЗАДАЧІ

4.1. Головна форма програми

Програма "Кулінарні рецепти" була створена за допомогою сучасної мови програмування C# в середовищі Visual Studio 2019 [5-8]. Для розробки використовувався Windows Forms. Усі візуальні елементи програми базуються на класі Form. Головний інтерфейс програми можна побачити на рис. 4.1.

Рис. 4.1. Головна форма програми

При реалізації головного меню передбачено наступні опції [додаток А]:

1. Довідники.
2. Ведення даних.
3. Формування рецепта
4. Звіт по рецептам
5. Налаштування.
6. Діаграми в HTML.
7. Про програму.
8. Вихід.

Опція «Довідники» складається з чотирьох пунктів меню:

1. «Категорія страв» – ведення довідника категорії страв.

2. «Категорія продуктів» – ведення довідника категорії продуктів.

3. «Країни рецептів» – ведення інформації країн з рецептами.

4. «Умови заходів» – ведення інформації по заходам (весілля, день народження тощо).

Опція «Ведення даних» складається з двох пунктів меню:

1. «Робота зі стравами» – ведення інформації по стравам окремої вибраної категорії.

2. «Робота з продуктами» ведення інформації по продуктам окремої вибраної категорії.

Опція «Формування рецепта» призначена для формування рецепту на основі існуючих даних та запис його до бази даних.

Опція «Звіт по рецептам» призначена для формування звітної документації по вибраним рецептам в формат MS Excel.

Опція «Налаштування» призначена для зміни кольору та шрифту вхідних та вихідних даних, зокрема звітів.

При натисканні опції «Про програму» отримується інформація про розробника програмного забезпечення (рис. 4.2.).

При виборі опції «Вихід» відбувається вихід з програми із збереженням всіх вибраних налаштувань.

На основній формі програми відбувається з’єднання з базою даних "Кулинарія.mdb" за допомогою об'єкту OleDBConnection. Після цього цей об'єкт передається до інших форм програми для використання.

Рис. 4.2. Форма «Про програму»

4.2. Розробка програмного забезпечення та тестування задачі

Давайте розглянемо клас Form2, який використовується для управління довідниками, такими як категорія страв, категорія продуктів, країни рецептів та умови заходів. На рис. 4.3 наведено приклад роботи з формою для управління довідником категорії страв.

У верхній частині форми розташований елемент Label, в середині – елемент роботи з таблицями DataGridView, а у нижній частині – рядок навігації даних (клас BindingNavigator). Слід зауважити, що до рядка навігації додано кнопку "Оновити" для оновлення даних в таблиці бази даних, а також кнопки із значком "xls" для експорту даних в MS Excel та кнопки із значком "html" для експорту даних у формат html.

У формі також присутня цілочисельна змінна kod_table, яка відповідає

за завантаження конкретного довідника (SQL-запиту). Якщо значення змінної kod_table дорівнює одиниці, то завантажується довідник з інформацією щодо категорій страв - "CategoriesDish".

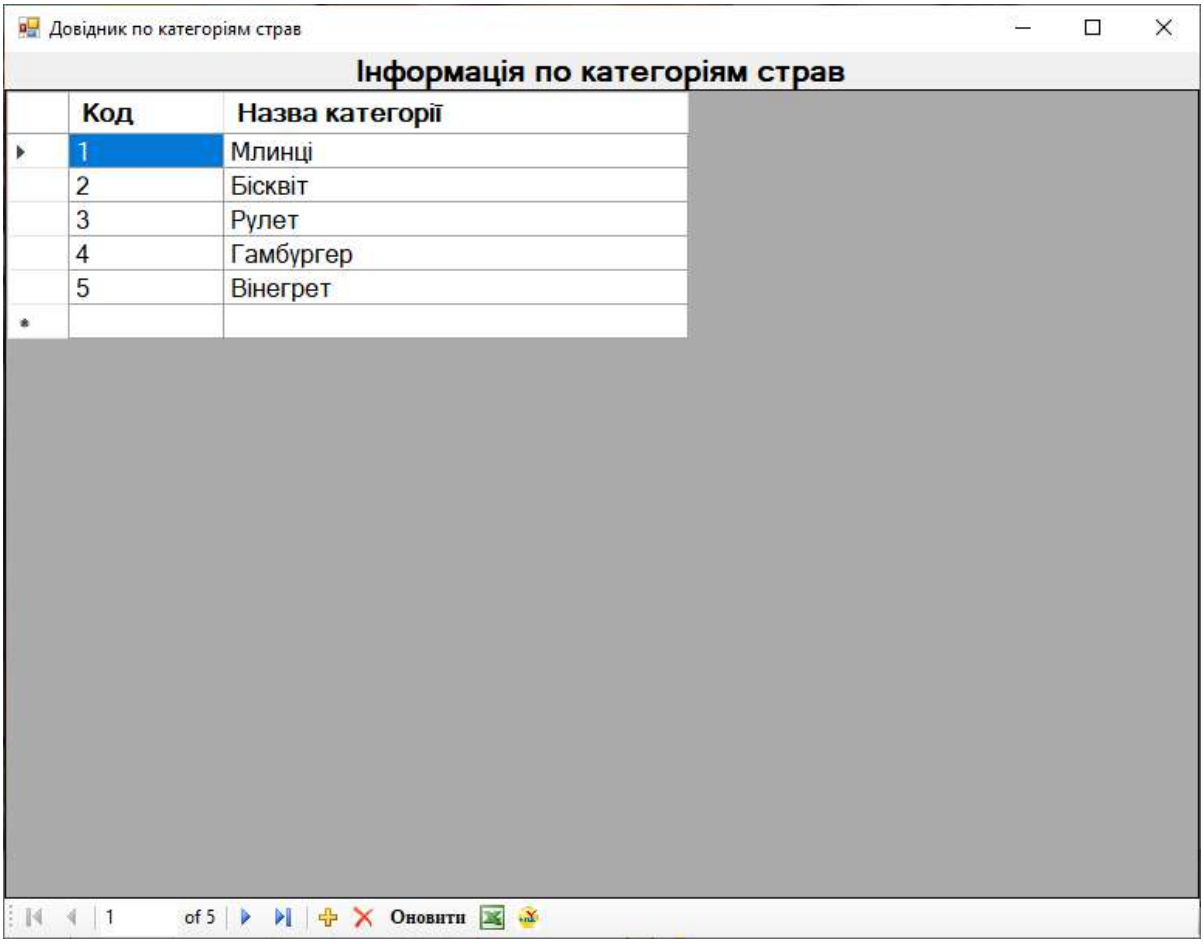

Рис. 4.3. Ведення довідника категорії страв

Наведемо список та коротку характеристику основних функцій класу Form2.

1. Form2_Load – функція завантаження форми.
2. toolStripButton1_Click – функція оновлення даних.
3. toolStripButton2_Click – функція експорту даних в MS Excel.
4. toolStripButton3_Click – функція експорту даних в HTML.

Приведемо код функції Form2_Load.

```
//Завантаження форми
        private void Form2_Load(object sender, EventArgs e)
        {
            try
            {
                if (kod_table == 1)
```

```
                {
                    Text = "Довідник по категоріям страв";
                    label1.Text = "Інформація по категоріям страв";
                    da = new OleDbDataAdapter("Select ID, Name from CategoriesDish",
conn);
                    bulder = new OleDbCommandBuilder(da);
                    ds = new DataSet();
                    DataTableMapping datam = da.TableMappings.Add("Table", "Категорія
страв");
                    datam.ColumnMappings.Add("ID", "Код");
                    datam.ColumnMappings.Add("Name", "Назва категорії");
                }
                if (kod_table == 2)
                {
                    Text = "Довідник по категоріям продуктів";
                    label1.Text = "Інформація по категоріям продуктів";

                    da = new OleDbDataAdapter("Select ID, Name from CategoriesProduct",
conn);
                    bulder = new OleDbCommandBuilder(da);
                    ds = new DataSet();

                    DataTableMapping datam = da.TableMappings.Add("Table", "Категорія
продуктів");
                    datam.ColumnMappings.Add("ID", "Код");
                    datam.ColumnMappings.Add("Name", "Назва категорії");

                }

                if (kod_table == 3)
                {
                    Text = "Довідник по країнам для рецептів";
                    label1.Text = "Інформація по країнам для рецептів";

                    da = new OleDbDataAdapter("Select ID, Name from Country", conn);
                    bulder = new OleDbCommandBuilder(da);
                    ds = new DataSet();

                    DataTableMapping datam = da.TableMappings.Add("Table", "Країни");
                    datam.ColumnMappings.Add("ID", "Код");
                    datam.ColumnMappings.Add("Name", "Найменування");
                }
                if (kod_table == 4)
                {
                    Text = "Довідник по умовам (весілля, день народження)";
                    label1.Text = "Інформація по умовам (весілля, день народження)";

                    da = new OleDbDataAdapter("Select ID, Name from Terms", conn);
                    bulder = new OleDbCommandBuilder(da);
                    ds = new DataSet();
                    DataTableMapping datam = da.TableMappings.Add("Table", "Умови");
                    datam.ColumnMappings.Add("ID", "Код");
                    datam.ColumnMappings.Add("Name", "Назва");
                }
                da.Fill(ds);
                bindingSource1.DataSource = ds.Tables[0];
                dataGridView1.DataSource = bindingSource1;
                bindingNavigator1.BindingSource = bindingSource1;

                //Установка по першим двом полям
                dataGridView1.Columns[0].ReadOnly = true;
                dataGridView1.Columns[1].Width = 300;
            }
            catch (Exception ex)
            {
```

```
            MessageBox.Show(ex.Message);
        }
    }
```

У цій функції створюється об'єкт класу адаптера da для взаємодії з однією з таблиць бази даних: "CategoriesDish", "CategoriesProduct", "Country", "Terms", в залежності від значення змінної kod_table. З використанням цього об'єкту створюється набір записів, який прив'язується до елементу DataGridView. Після цього можна повністю редагувати дані таблиці - додавати, видаляти, змінювати записи. Проте варто зауважити, що всі зміни відбуваються лише у елементі DataGridView. Для того щоб відображені зміни відобразилися у базі даних, необхідно натиснути кнопку оновлення даних. За формування запитів на оновлення (UPDATE), видалення (DELETE) та додавання (INSERT) даних відповідає клас OleDBCommandBuilder.

При натисненні на кнопку «xls» – інформацію буде виведено у MS Excel, звідки її можна роздрукувати у зручному для користувача вигляді (рис. 4.4).

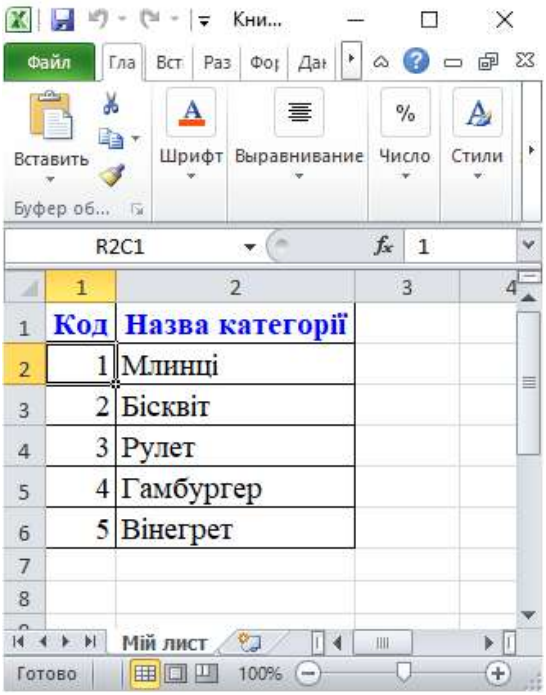

Рис. 4.4. Експорт даних до MS Excel

При натисненні на кнопку «html» – інформацію буде виведено у формат html (рис. 4.5). Код html-сторінки виводиться в розробленій програмі

– у формі Form4. У якості елементу керування для перегляду web-сторінок використано елемент WebBrowser.

Рис. 4.5. Експорт даних в HTML

Наступним етапом є проектування класу Form3 де відбувається ведення інформації по стравам або продуктам конкретної категорії. Так як таблиці бази даних пов'язані відношенням один до багатьох, спочатку необхідно вибрати конкретну категорію, а потім по ній сформувати SQL-запит. З цією метою спроектовано клас Name_List, який вміщує базову інформацію для будь-якого списку – рядкове значення та значення ключового ідентифікатору – ID.

Наведемо код класу Name_List.

```
//Клас для заповнення списку
public class Name_List
{
    public int ID;
    public string Name;

    public Name_List(int ID, string Name)
    {
        this.ID = ID;
        this.Name = Name;
    }

    public override string ToString()
    {
```

```
            return Name;
        }
    }
```

Наведемо результат виконання форми Form3 для ведення даних інформації по стравам вибраної категорії (рис. 4.6.).

Наведемо список та коротку характеристику основних функцій класу Form3.

1. Form3_Load – функція завантаження форми.
2. comboBox1_SelectedIndexChanged – вибір групи виконавця зі списку.
3. toolStripButton1_Click – функція оновлення даних.
4. toolStripButton2_Click – функція експорту даних в MS Excel.
5. toolStripButton3_Click – функція експорту даних в HTML.

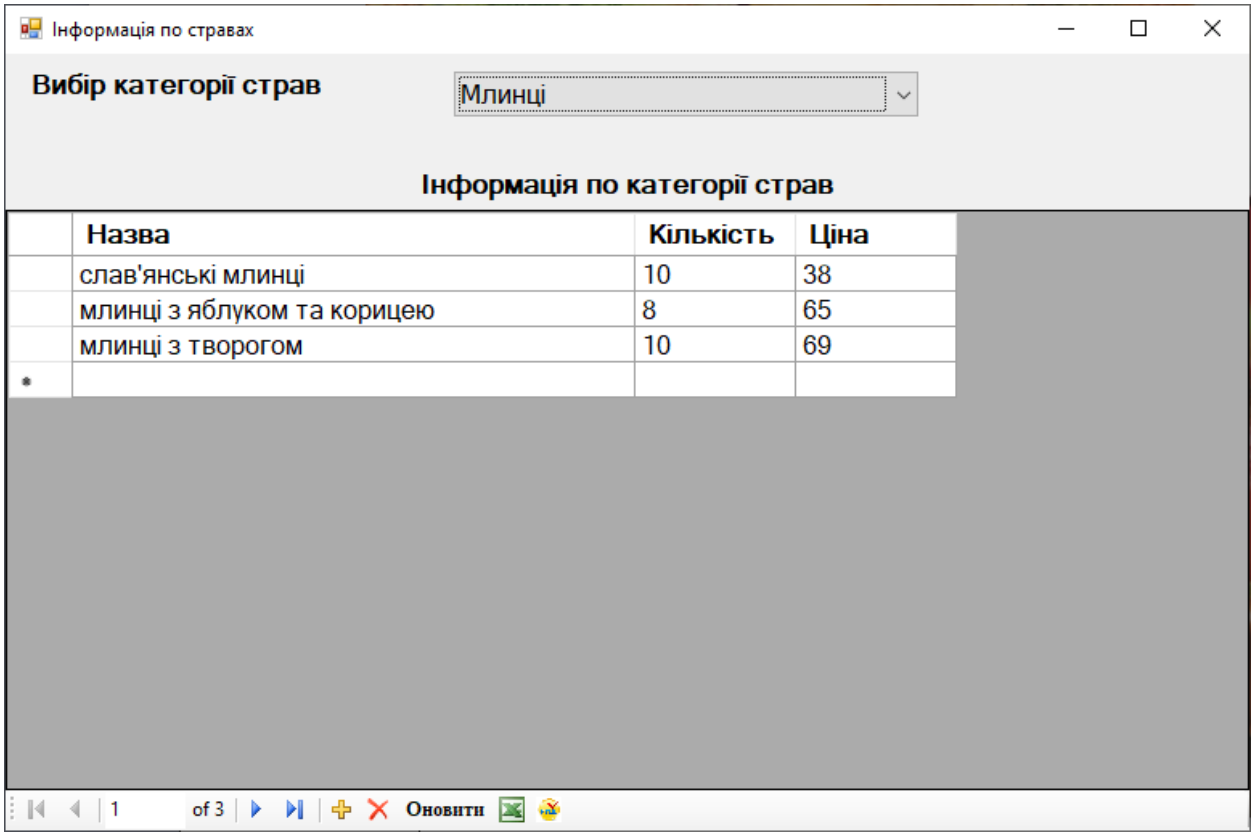

Рис. 4.6. Проектування форми Form3

Приведемо код функції Form3_Load.

```
//Завантаження форми
        private void Form3_Load(object sender, EventArgs e)
        {
```

```
            try
            {
                // Заповнення списку SQL - запитом
                string zapros="";

                if (kod_table == 1)
                {
                    Text = "Інформація по стравах";
                    label1.Text = "Вибір категорії страв";
                    label2.Text = "Інформація по категорії страв";
                    zapros = "Select ID, Name From CategoriesDish";
                }

                if (kod_table == 2)
                {
                    Text = "Інформація по продуктах";
                    label1.Text = "Вибір категорії продуктів";
                    label2.Text = "Інформація по категорії продуктів";
                    zapros = "Select ID, Name From CategoriesProduct";
                }

                cmd = new OleDbCommand(zapros, conn);
                datar = cmd.ExecuteReader();
                while (datar.Read())
                    comboBox1.Items.Add(new Name_List(Convert.ToInt32(datar[0]),
datar[1].ToString()));

                comboBox1.SelectedIndex = 0;
                datar.Close();
                cmd.Dispose();
                ////////////////////////////////
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }
```

У даній функції відбувається заповнення списку combobox1 інформацією по категоріям страв з використанням класу Name_List. Для виконання запиту використовується об’єкт OleDBCommand. Результат виконання записується в об’єкт для читання OleDBDataReader, який застосовується для заповнення елементів керування та призначений тільки для читання записів.

Далі наведемо код функції comboBox1_SelectedIndexChanged, яка спрацьовує при виборі відповідної категорії страв.

```
//Вибір елемента списку
        private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
        {
            try
            {
                string str = "";
```

```
                if (kod_table == 1) str = "Select ID, Name, Kol, Cena from Dishes where
";
                if (kod_table == 2) str = "Select ID, Name, Kol, Calorie, Belki, Zhiry,
Uglev from Products where ";
                str += "ID_cat = " + ((Name_List)comboBox1.SelectedItem).ID;
                da = new OleDbDataAdapter(str, conn);
                bulder = new OleDbCommandBuilder(da);
                ds = new DataSet();
                DataTableMapping datam = null;
                if (kod_table == 1)
                {
                    datam = da.TableMappings.Add("Table", "Страви");
                    datam.ColumnMappings.Add("ID", "Код");
                    datam.ColumnMappings.Add("Name", "Назва");
                    datam.ColumnMappings.Add("Kol", "Кількість");
                    datam.ColumnMappings.Add("Cena", "Ціна");
                }

                if (kod_table == 2)
                {
                    datam = da.TableMappings.Add("Table", "Продукти");
                    datam.ColumnMappings.Add("ID", "Код");
                    datam.ColumnMappings.Add("Name", "Найменування");
                    datam.ColumnMappings.Add("Kol", "Фасування, г");
                    datam.ColumnMappings.Add("Calorie", "Калорійність");
                    datam.ColumnMappings.Add("Belki", "Білки");
                    datam.ColumnMappings.Add("Zhiry", "Жири");
                    datam.ColumnMappings.Add("Uglev", "Вуглеводи");
                }
               da.Fill(ds);
                bindingSource1.DataSource = ds.Tables[0];
                dataGridView1.DataSource = bindingSource1;
                bindingNavigator1.BindingSource = bindingSource1;
                dataGridView1.Columns[0].Visible = false;
                dataGridView1.Columns[1].Width = 350;
                bindingSource1.Position = 0;
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }
```

При виборі категорії страв зі списку формується запит по конкретному ідентифікатору. Результат відповідного запиту по категорії страв прив’язується до елементу DataGridView з подальшими можливостями редагування.

Слід сказати, що окрім розроблених форм мають місце універсальні функції роботи з даними.

Перевагою експорту в HTML є те, що сформований html-файл відображається у власному середовищі за допомогою елементу керування WebBrowser.

На наступному етапі розглянемо безпосередньо формування рецепту

окремої страви.

Розглянемо форму для формування рецепту страви Form5 (рис. 3.7).

Як видно з рис. 4.7. для формування рецептів страв треба вибрати країну, категорію страви, страву, категорію продуктів, а також продукти із списку, за якими буде формуватися окремий рецепт.

При натисканні на кнопку «Формування рецепта» даний рецепт записується до бази даних.

Розглянемо форму звітів по рецептам страв Form6 (рис. 4.8).

Як видно з рис. 4.8. для формування звітів за рецептами страв необхідно обрати номер попередньо-сформованого рецепту.

При натисканні на кнопку «Експорт в XLS» даний рецепт відкривається у середовищі MS Excel. Це є зручним формуванням звітів, так як з середовища MS Excel можна змінити та роздрукувати довільну інформацію, представлену у вигляді таблиці.

Результат виконання форми «Звіт по рецептам» приведено на рис. 4.9.

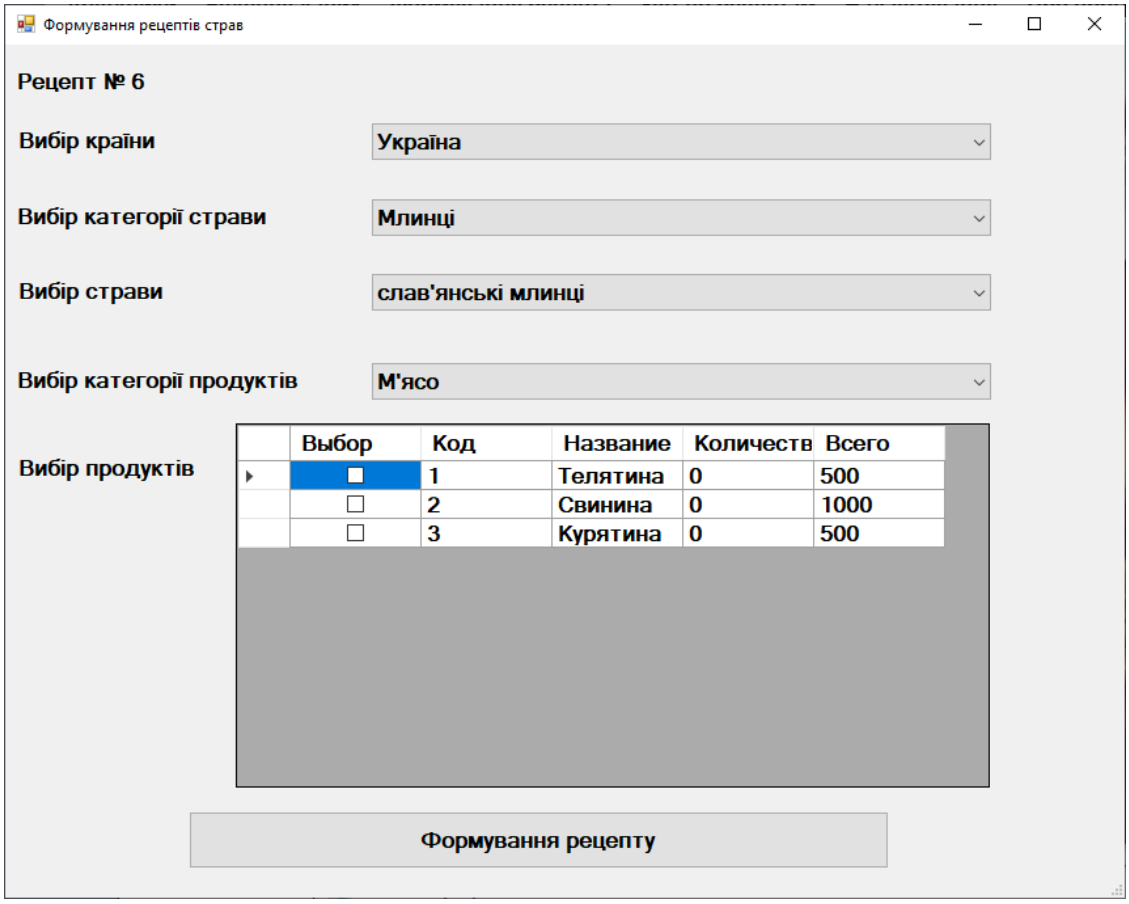

Рис. 4.7. Результат роботи з формою Form5 «Формування рецептів страв»

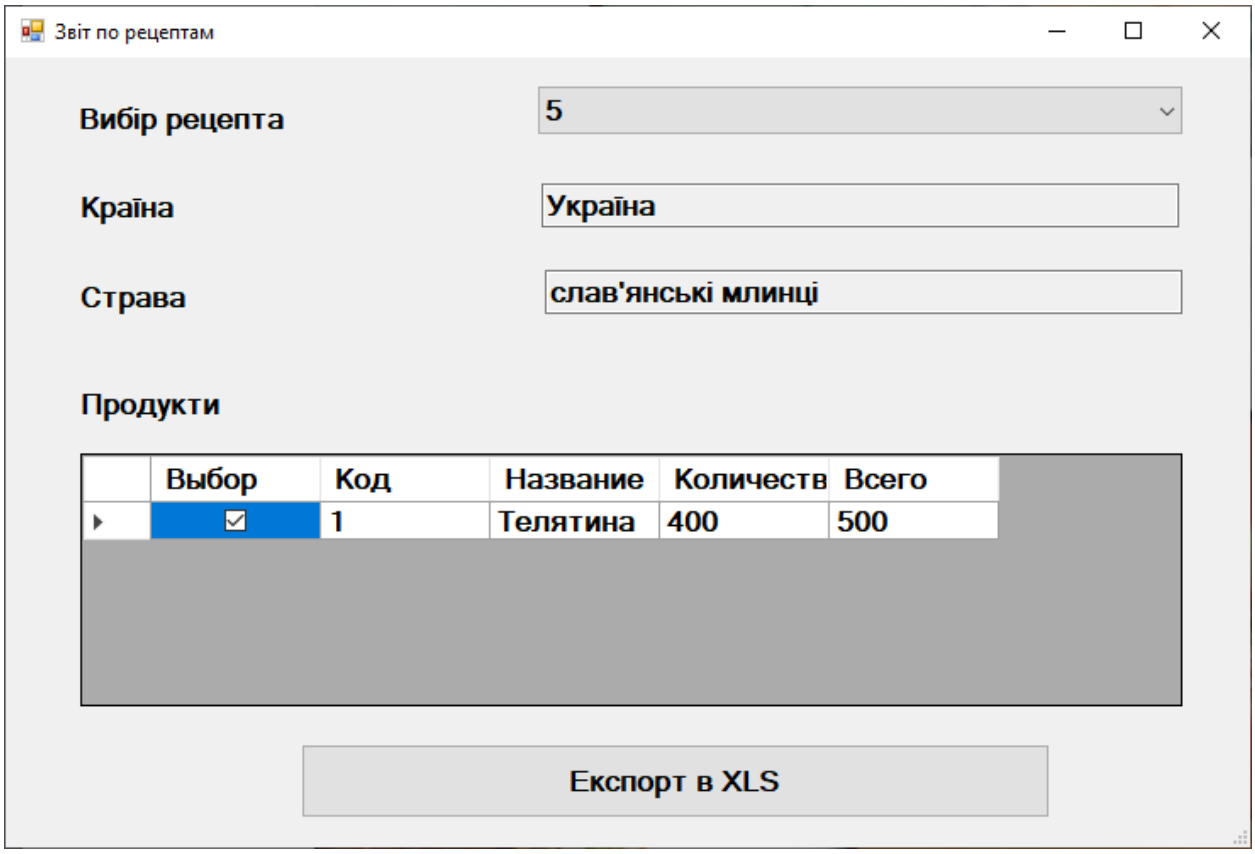

Рис. 4.8. Результат роботи з формою Form6 «Звіт по рецептам»

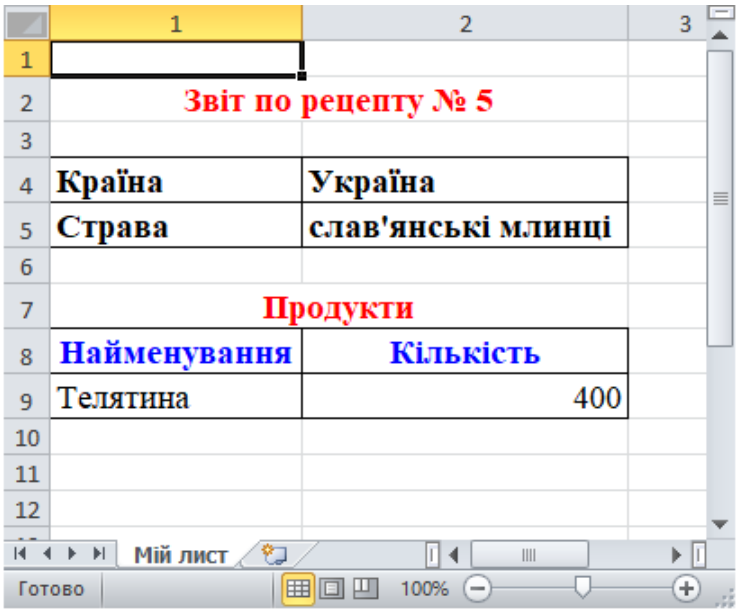

Рис. 4.9. Формування звіту по рецепту номер 5

Важливим етапом при розробці будь-якого програмного забезпечення є робота з налаштуваннями. У нашій програмі за це відповідає меню «Налаштування». Через дане меню можна змінити кольори та шрифт звітної інформації.

Змінити шрифт можливо натиснувши пункт меню «Вибір шрифту». У відкритому діалозі зміни можливо здійснити всі необхідні зміни, що спростили б використання запропонованої програми (рис. 4.10).

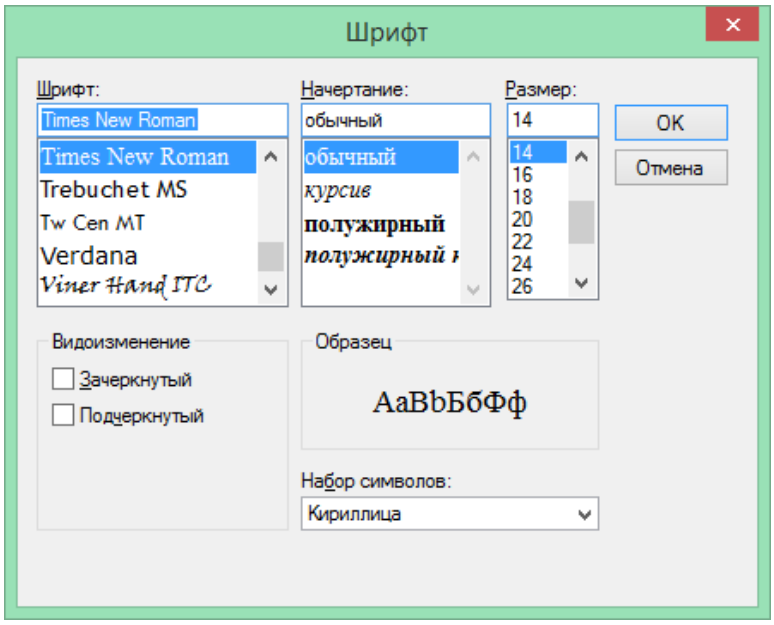

Рис. 4.10. Вікно вибору шрифту

Як видно з наданого малюнка, можливо змінити текст використовуючи курсив або виділити його жирним.

Для обрання необхідного кольору тексту, необхідно натиснути пункт меню «Вибір кольору» (рис. 4.11). Змінювати текст можливо також, використовуючи палітру, а не стандартні кольори.

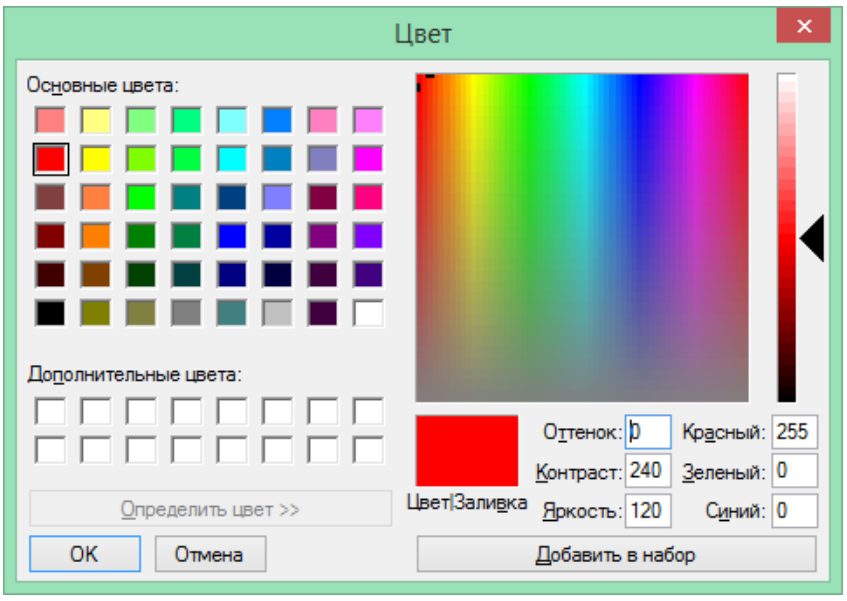

Рис. 4.11. Вікно вибору кольору

4.3. Опис програми кулінарних рецептів «Cooking_3D»

За допомогою мови Visual C++ та використання бібліотек OpenGL було створене 3D зображення полігону та гістограми, для того щоб наочно проаналізувати калорійність продуктів. Була передбачена робота зі світлом, виведення тіні стовпчиків, виведення тексту, накладення текстур на землю, градієнтна заливка.

Наведемо фрагмент коду зчитування інформації з бази даних, використовуючи технологію ADO.

```
// Формування рядка підключення до БД

	CString str_con = L"Provider=Microsoft.Jet.OLEDB.4.0;Data Source=Кулинария.mdb";

	try
	{

		pConn->Open((_bstr_t)str_con, L"", L"", 0);
		//Заповнення масивів та закриття набору даних
		pRecordset = pConn->Execute("Select Name, Calorie From Products", 0,
adCmdText);
		FieldsPtr fields = pRecordset->Fields;
		pRecordset->MoveFirst();
		long i = 0;
		while (!pRecordset->ADO_EOF)
		{
			strcpy_s(name[i], (char*)(_bstr_t)fields->GetItem(0L)->GetValue());
			calorie[i] = fields->GetItem(1L)->GetValue();
			pRecordset->MoveNext();
			i++;
		}

		count = i;

		pRecordset->Close();
		pRecordset.Release();

	////////////////////////////////////////////////////////////////////////////////
///////////

		pConn->Close();
		pConn.Release();
	}

	catch (_com_error &ce)
	{
		ErrMessage(ce);
		if (pConn->GetState())pConn->Close();
		return;
	}
```

На рис. 4.12. виведено полігон по калорійності продуктів.

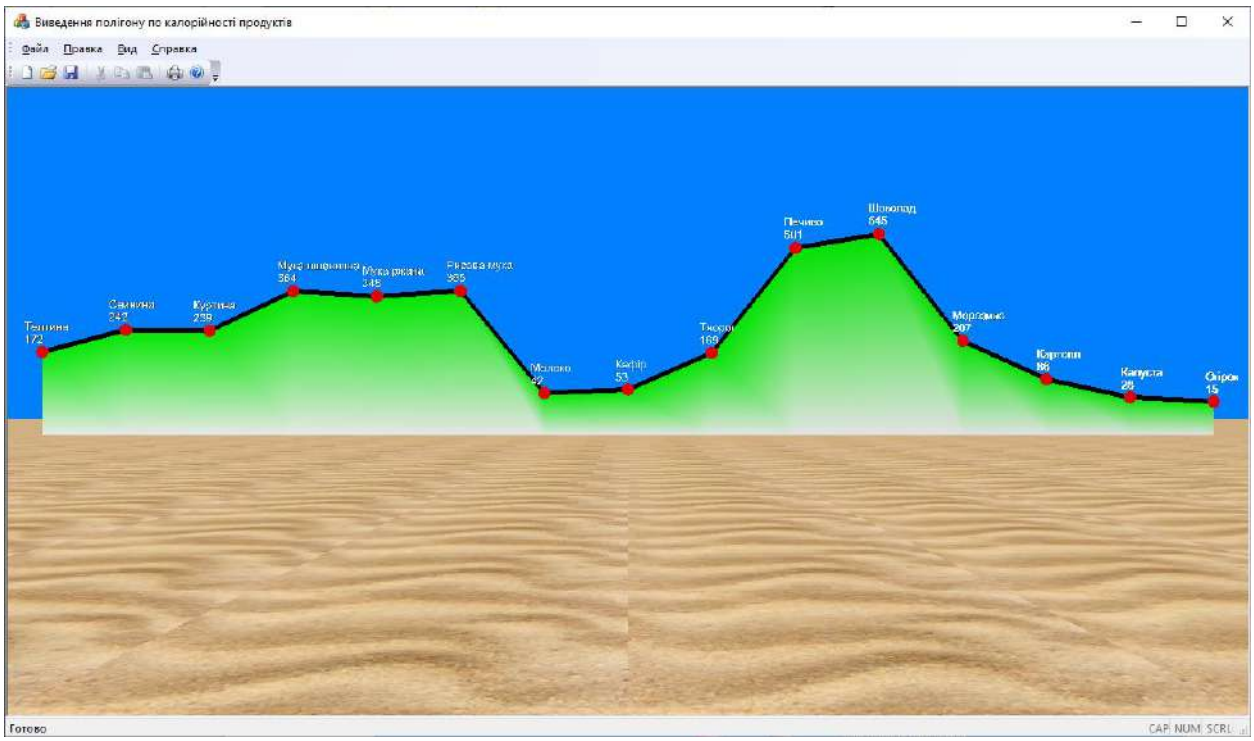

Рис. 4.12. Виведення полігону по калорійності продуктів

На рис. 4.13. виведено гістограму по калорійності продуктів.

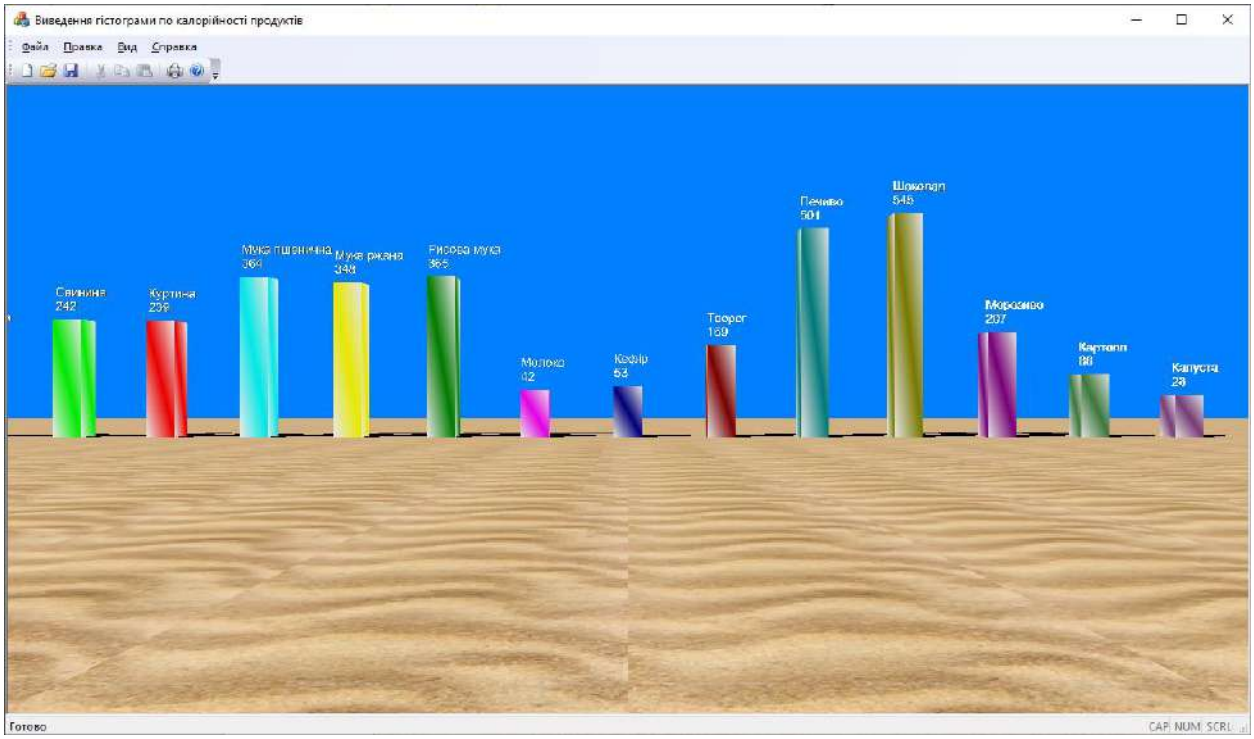

Рис. 4.13. Виведення гістограми по калорійності продуктів

4.4. Опис скриптів для формування діаграм у форматі HTML

У якості web-частини даного програмного забезпечення вибрано автоматичне формування скриптів на основі елементу керування chart.js. З цією метою розроблено універсальну функцію, до якою треба передати два масиви вхідних параметрів – числові параметри по яким будується діаграма, а також рядкові параметри для виведення підказок та легенди стовпчиків.

Наведемо код даної функції:

```
//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)}]";
        }
```

Дана функція є універсальною і дозволяє автоматично формувати Java-скрипти по даним які беруться з бази даних та відповідна інформація відразу відкривається з браузеру. Це дозволяє користуватися статистикою проданих дисків не тільки з десктоп-додатку, а і з мобільних пристроїв.

Гістограма в HTML статистики калорійності продуктів продемонстрована на рис. 4.14.

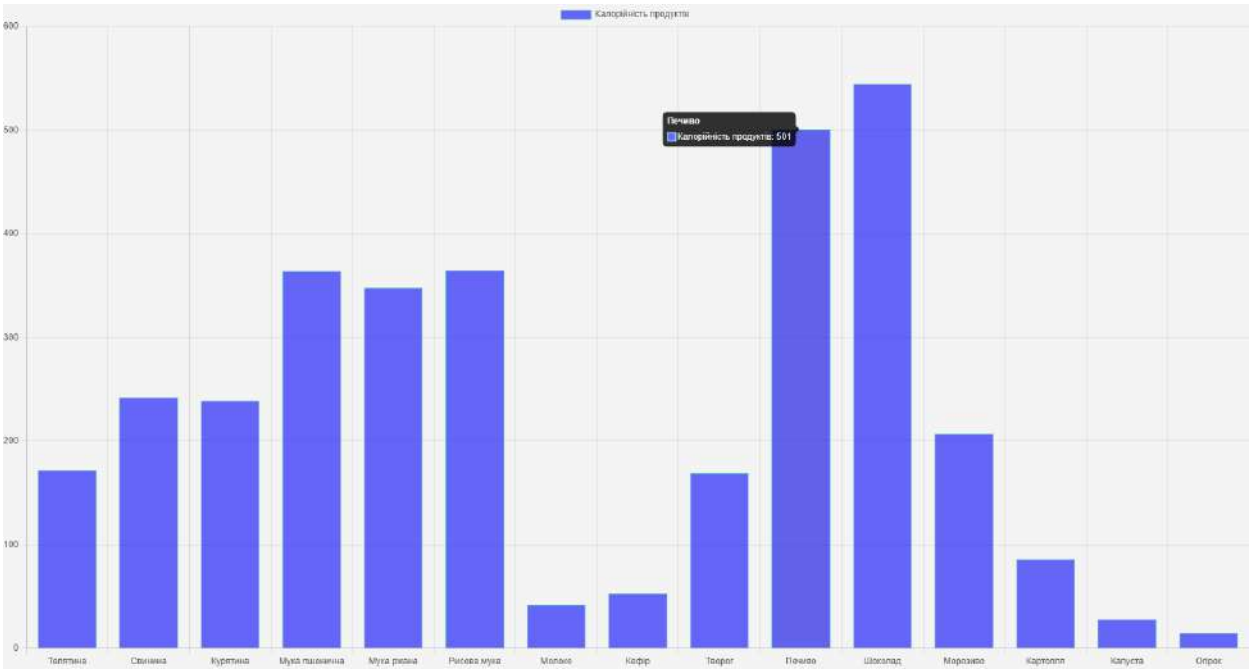

Рис. 4.14. Гістограма в HTML по калорійності продуктів

Крім того, передбачено скрипт для виведення кругової діаграми за допомогою методу GenerateChartHTMLPie, код якого наведено нижче.

```
//Скрипт для генерації кругової діаграми
        public static void GenerateChartHTMLPie(string fileName, string pageTitle, string
chartTitle, string[] labels, int[] data, string[] colors)
        { string labelsArray = "['" + string.Join("', '", labels) + "']";
            string dataArray = "[" + string.Join(", ", data) + "]";
            string colorsArray = "['" + string.Join("', '", colors) + "']";
            string htmlContent = $@"<!DOCTYPE html>
<html lang='uk'>
<head>
    <meta charset='UTF-8'>
    <meta name='viewport' content='width=device-width, initial-scale=1.0'>
    <title>{pageTitle}</title>
    <script src='https://cdn.jsdelivr.net/npm/chart.js'></script>
    <style>
        body {{
            font-family: Arial, sans-serif;
            text-align: center;
            background-color: #f4f4f4;
            margin: 0;
            display: flex;
            justify-content: center;
            align-items: center;
            height: 100vh;
        }}
        canvas {{
            width: 90vmin !important;
            height: 90vmin !important;
        }}
    </style>
</head>
<body>
    <center><h2>{chartTitle}<h2><center>
    <canvas id='populationChart'></canvas>
    <script>
        const ctx = document.getElementById('populationChart').getContext('2d');
        new Chart(ctx, {{
            type: 'pie',
            data: {{
                labels: {labelsArray},
                datasets: [{{
                    label: '{chartTitle}',
                    data: {dataArray},
                    backgroundColor: {colorsArray},
                    borderWidth: 1
                }}]
            }},
            options: {{
                responsive: true,
                maintainAspectRatio: true,
                plugins: {{
                    legend: {{
                        position: 'bottom'
                    }}
                }}
            }}
        }});
    </script>
</body>
</html>";
            File.WriteAllText(fileName, htmlContent);
}
```

Відповідну кругову діаграму по калорійності продуктів наведено на рис. 4.15.

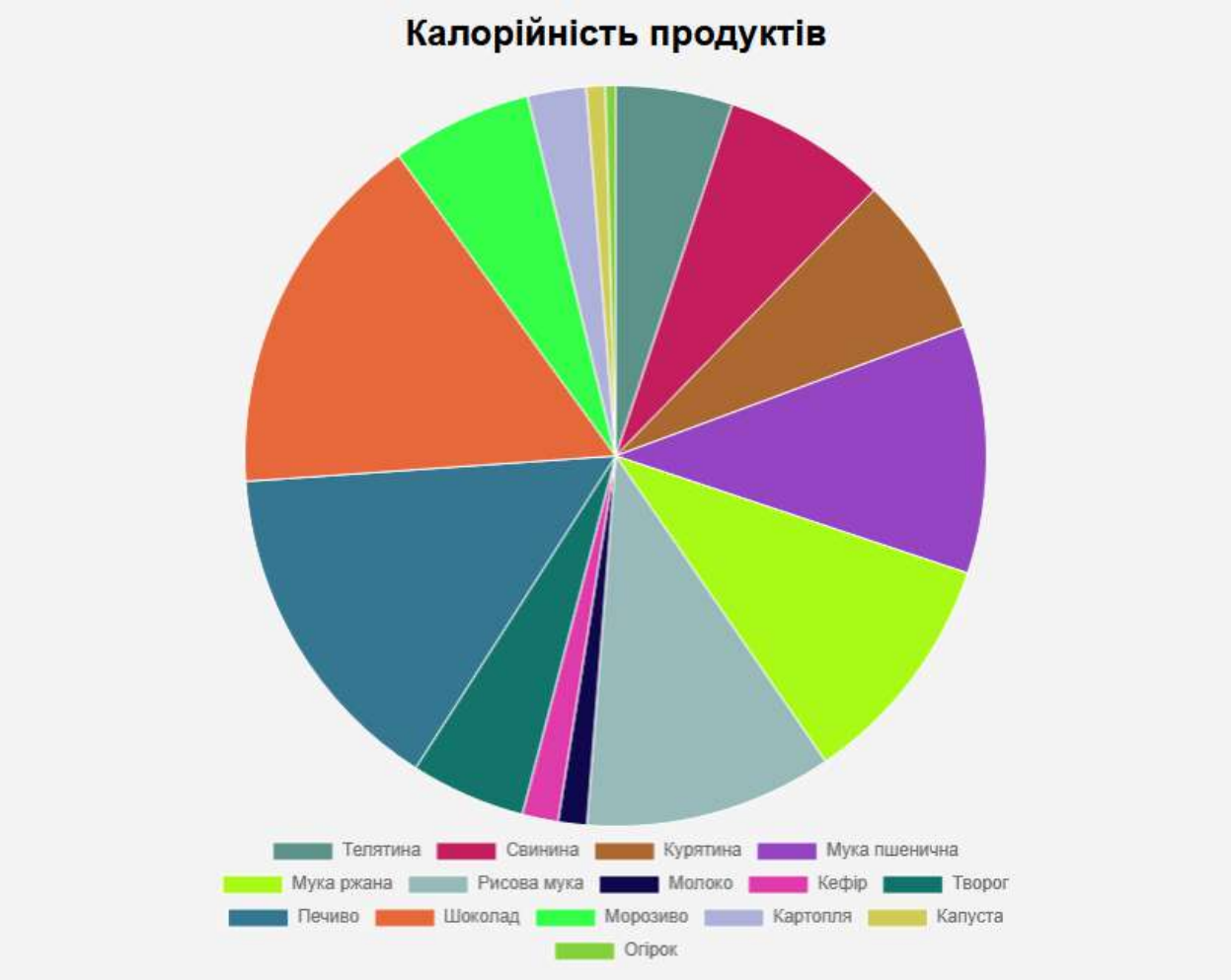

Рис. 4.15. Кругова діаграма в HTML по калорійності продуктів

Висновки до розділу 4

У даному розділі наводиться розробка програмного забезпечення кулінарного довідника для формування рецептів на мові C#. Відповідне програмне забезпечення дозволяє автоматизувати процес формування рецептів та вирішити наступні задачі: ведення даних по стравам; ведення даних по продуктам; формування рецептів, запис їх до бази даних та автоматичне формування звітів; автоматизоване формування звітів та їх експорт в MS Excel та HTML.

ВИСНОВКИ

У ході виконання дипломної роботи були виконані наступні задачі:

1. Розроблено програмне забезпечення кулінарних рецептів з автоматизацією формування рецептів по обраним продуктам.

2. Використано основні класи універсальної технології ADO.Net для обробки баз даних.

3. Розроблено та створено візуальні класи Form1, Form2, Form3, Form4, Form5, Form6, Form7.

4. Розроблено універсальні функції експорту даних у MS Excel та HTML.

Відповідне програмне забезпечення дозволяє автоматизувати процес формування рецептів та вирішити наступні задачі:

– ведення даних по стравам;

– ведення даних по продуктам;

– формування рецептів, запис їх до бази даних та автоматичне формування звітів;

– автоматизоване формування звітів та їх експорт в MS Excel та HTML.

Програмний комплекс «Cooking_3D», розроблений на мові C++ з використанням відкритої графічної бібліотеки OpenGL, дає можливість наочного подання інформації, у вигляді графіків гістограми та полігону. За допомогою цих графіків надається можливість наочно аналізувати інформацію про калорійність тих чи інших продуктів при формуванні рецептів.

Застосування універсальних Java-скриптів на основі елементу керування Chart.js дають можливість гнучко передавати дані з бази до створюваного html-файлу та користуватися ним в будь-яких браузерах. Це дозволило зробити гістограму та кругову діаграму в html для аналізу калорійності продуктів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Агуров, Павел C#. Сборник рецептов / Павел Агуров. - М.: "БХВ-Петербург", 2012. - 432 с.

2. Албахари, Джозеф C# 3.0. Справочник / Джозеф Албахари, Бен Албахари. - М.: БХВ-Петербург, 2012. - 944 с.

3. Албахари, Джозеф C# 3.0. Справочник / Джозеф Албахари, Бен Албахари. - М.: БХВ-Петербург, 2013. - 944 с.

4. Альфред, В. Ахо Компиляторы. Принципы, технологии и инструментарий / Альфред В. Ахо и др. - М.: Вильямс, 2015. - 266 с.

5. Бишоп, Дж. C# в кратком изложении / Дж. Бишоп, Н. Хорспул. - М.: Бином. Лаборатория знаний, 2013. - 472 с.

6. Вагнер, Билл C# Эффективное программирование / Билл Вагнер. - М.: ЛОРИ, 2013. - 320 с.

7. Зиборов, В.В. Visual C# 2012 на примерах / В.В. Зиборов. - М.: БХВ-Петербург, 2013. - 480 с.

8. Зиборов, Виктор Visual C# 2010 на примерах / Виктор Зиборов. - М.: "БХВ-Петербург", 2011. - 432 с.

9. Ишкова, Э. А. Самоучитель C#. Начала программирования / Э.А. Ишкова. - М.: Наука и техника, 2013. - 496 с.

10. Касаткин, А. И. Профессиональное программирование на языке си. Управление ресурсами / А.И. Касаткин. - М.: Высшая школа, 2012. - 432 с.

11. Лотка, Рокфорд C# и CSLA .NET Framework. Разработка бизнес-объектов / Рокфорд Лотка. - М.: Вильямс, 2010. - 816 с.

12. Мак-Дональд, Мэтью Silverlight 5 с примерами на C# для профессионалов / Мэтью Мак-Дональд. - М.: Вильямс, 2013. - 848 с.

13. Марченко, А. Л. Основы программирования на C# 2.0 / А.Л. Марченко. - М.: Интернет-университет информационных технологий, Бином. Лаборатория знаний, 2011. - 552 с.

14. Подбельский, В. В. Язык C#. Базовый курс / В.В. Подбельский. - М.:

Финансы и статистика, Инфра-М, 2011. - 384 с.

15. Прайс, Джейсон Visual C# 2.0. Полное руководство / Джейсон Прайс, Майк Гандэрлой. - М.: Век +, Корона-Век, Энтроп, 2010. - 736 с.

16. Рихтер, Джеффри CLR via C#. Программирование на платформе Microsoft .NET Framework 4.0 на языке C# / Джеффри Рихтер. - М.: Питер, 2013. - 928 с.

17. Смоленцев, Н. К. MATLAB. Программирование на Visual C#, Borland JBuilder, VBA (+ CD-ROM) / Н.К. Смоленцев. - М.: ДМК Пресс, 2011. - 456 с.

18. Троелсен, Эндрю Язык программирования C# 5.0 и платформа .NET 4.5 / Эндрю Троелсен. - М.: Вильямс, 2015. - 486 с.

19. Троелсен, Эндрю Язык программирования C# 2008 и платформа .NET 3.5 / Эндрю Троелсен. - М.: Вильямс, 2010. - 370 с.

20. Фримен, Адам ASP.NET MVC 3 Framework с примерами на C# для профессионалов / Адам Фримен , Стивен Сандерсон. - М.: Вильямс, 2011. - 672 с.

21. Дж.Рихтер. CLR via C# . Программирование на платформе Microsoft .NET Framework 4.5 на языке C#. 4-е изд./Дж.Рихтер – Питер, 2013. - 896с.

22. Дж.Рихтер. CLR via C#. Программирование на платформе Microsoft .NET Framework 2.0 на языке C#, 2-е изд/ Дж.Рихтер - Питер, 2007. - 656 с.

23. Зеленков Ю. Введение в базы данных / Ю. Зеленков – СПб.: Питер, 2003

24. Кузнецов С.Д. Основы современных баз данных. 2-е издание испр. / С.Д.Кузнецов,– М.:Интернет-Университет Информационных Технологий; БИНОМ. Лаборатория знаний– 2007. – 484с.

ДОДАТКИ

Додаток А

Лістинг головної форми програми Form1

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.OleDb;
using System.Drawing;
using System.Drawing.Drawing2D;
using System.Linq;
using System.Text;
using System.IO;
using System.Windows.Forms;
using Excel = Microsoft.Office.Interop.Excel;

namespace Kursova1
{
    public partial class Form1 : Form
    {

        //Поключение к БД
        OleDbConnection conn;

        //Настройки
        public static Color cvet_zag;
        public static Color cvet_hapka;
        public static Color cvet_text;
        public static Font font_text;

        int kod_zap = 0;


        //Конструктор
        public Form1()
        {
            InitializeComponent();

            //Формирование объекта Connection
            try
            {
                string source = "Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=Кулинария.mdb";
                conn = new OleDbConnection(source);
                conn.Open();
            }

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

            //Считывание настроек
            try
            {
                BinaryReader bb = new BinaryReader(new FileStream("кулинария.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
            {
                MessageBox.Show("Файл кулинария.ini не найден!\nБудут приняты значения по
умолчанию");
            }


            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;
        }

        //Категория блюд
        private void категорияБлюдToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Form2 ff = new Form2();
            ff.conn = conn;
            ff.kod_table = 1;
            ff.ShowDialog();
        }

        //Категория продуктов
        private void категорияПродуктовToolStripMenuItem_Click(object sender, EventArgs
e)
        {
            Form2 ff = new Form2();
            ff.conn = conn;
            ff.kod_table = 2;
            ff.ShowDialog();
        }

        //Страны
        private void страныРецептовToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Form2 ff = new Form2();
            ff.conn = conn;
            ff.kod_table = 3;
            ff.ShowDialog();
        }

        //Условия
        private void условияМероприятийToolStripMenuItem_Click(object sender, EventArgs
e)
        {
            Form2 ff = new Form2();
            ff.conn = conn;
            ff.kod_table = 4;
            ff.ShowDialog();
        }

        //Выход
        private void выходToolStripMenuItem_Click(object sender, EventArgs e)
        {
            this.Close();
        }
```

```
        //Блюда
        private void работаСБлюдамиToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Form3 ff = new Form3();
            ff.conn = conn;
            ff.kod_table = 1;
            ff.ShowDialog();
        }

        //Продукты
        private void работаСПродуктамиToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Form3 ff = new Form3();
            ff.conn = conn;
            ff.kod_table = 2;
            ff.ShowDialog();
        }

        //Формирование рецепта
        private void формированиеРецептаToolStripMenuItem_Click(object sender, EventArgs
e)
        {
            Form5 ff = new Form5();
            ff.conn = conn;
            ff.ShowDialog();
        }



        //Универсальная функция экспорта в Excel произвольных данных
        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;
        }
```

Продовження додатку А

```
        //Универсальная функция экспорта в HTML произвольных данных
        // ф-я экспорта любой таблицы в файл html.
        // Вход:
        // filename - имя html-файла в который производится экспорт
        // dt - представление записей DataView
        // zaglav - заголовок таблицы
        // ff - задаваемый шрифт
        // cvetz - цвет заголовка таблицы
        // cveth - цвет шапки таблицы
        // cvett - цвет текста ячеек таблицы


        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("Экспорт таблицы");
                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");
```

Продовження додатку А

```
                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");

                //Экспорт шапки
                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");

                //Экспорт данных
                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;
```

```
            }
        }

        //Выбор шрифта
        private void выборШрифтаToolStripMenuItem_Click(object sender, EventArgs e)
        {
            fontDialog1.Font = font_text;
            if (fontDialog1.ShowDialog() == DialogResult.OK)
            {
                font_text = fontDialog1.Font;
                kod_zap = 1;
            }
        }

        //Выбор цвета заголовка таблицы
        private void цветЗаголовкаТаблицыToolStripMenuItem_Click(object sender, EventArgs
e)
        {
            colorDialog1.Color = cvet_zag;
            if (colorDialog1.ShowDialog() == DialogResult.OK)
            {
                cvet_zag = colorDialog1.Color;
                kod_zap = 1;
            }
        }

        //Выбор цвета шапки таблицы
        private void цветШапкиТаблицыToolStripMenuItem_Click(object sender, EventArgs e)
        {
            colorDialog1.Color = cvet_hapka;
            if (colorDialog1.ShowDialog() == DialogResult.OK)
            {
                cvet_hapka = colorDialog1.Color;
                kod_zap = 1;
            }
        }

        //Выбор цвета текста таблицы
        private void цветТекстаТаблицыToolStripMenuItem_Click(object sender, EventArgs e)
        {
            colorDialog1.Color = cvet_text;
            if (colorDialog1.ShowDialog() == DialogResult.OK)
            {
                cvet_text = colorDialog1.Color;
                kod_zap = 1;
            }
        }

        //Закрытие формы
        protected override void OnClosing(CancelEventArgs e)
        {

            if (kod_zap == 1)
            {
                DialogResult dr = MessageBox.Show("Настройки были изменены!\nСохранить
изменения на диск?", "Сохранение",
                MessageBoxButtons.YesNoCancel, MessageBoxIcon.Question);
                if (dr == DialogResult.Yes)
                {
                    BinaryWriter bb = new BinaryWriter(new FileStream("кулинария.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);
    }

    //Отчет по рецепту
    private void отчетыToolStripMenuItem_Click(object sender, EventArgs e)
    {
        Form6 ff = new Form6();
        ff.conn = conn;
        ff.ShowDialog();
    }

    //Диалог о программе
    private void оПрограммеToolStripMenuItem_Click(object sender, EventArgs e)
    {
        Form7 ff = new Form7();
        ff.ShowDialog();
    }

}

//Класс для заполнения списка
public class Name_List
{
    public int ID;
    public string Name;

    public Name_List(int ID, string Name)
    {
        this.ID = ID;
        this.Name = Name;
    }

    public override string ToString()
    {
        return Name;
    }
}

//Класс для заполнения продуктов
public class Name_Product
```

Продовження додатку А

```
    {
        public int id;
        public string name;
        public int kol;
        public int kol_t;
        public bool flag;

        public Name_Product(int id, string name, int kol, int kol_t, bool flag = false)
        {
            this.id = id;
            this.name = name;
            this.kol = kol;
            this.kol_t = kol_t;
            this.flag = flag;
        }

        public bool Выбор
        {
            get { return flag; }
            set { flag = value; }
        }


        public int Код
        {
            get { return id; }
        }

        public string Название
        {
            get { return name; }
        }

        public int Количество
        {
            get { return kol; }
            set { kol = value; }
        }

        public int Всего
        {
            get { return kol_t; }
        }

    }
}
```

Додаток Б

Лістинг головної форми програми Form2

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

namespace Kursova1
{
    public partial class Form2 : Form
    {
        public OleDbConnection conn;
        OleDbDataAdapter da;
        OleDbCommandBuilder bulder;
        DataSet ds;

        public int kod_table = 1;

        //Конструктор
        public Form2()
        {
            InitializeComponent();
        }

        //Загрузка формы
        private void Form2_Load(object sender, EventArgs e)
        {
            try
            {

                if (kod_table == 1)
                {

                    Text = "Справочник по категориям блюд";
                    label1.Text = "Информация по категориям блюд";

                    da = new OleDbDataAdapter("Select ID, Name from CategoriesDish",
conn);
                    bulder = new OleDbCommandBuilder(da);
                    ds = new DataSet();


                    DataTableMapping datam = da.TableMappings.Add("Table", "Категория
блюд");
                    datam.ColumnMappings.Add("ID", "Код");
                    datam.ColumnMappings.Add("Name", "Название категории");
                }

                if (kod_table == 2)
                {
                    Text = "Справочник по категориям продуктов";
                    label1.Text = "Информация по категориям продуктов";
```

Продовження додатку Б

```
                da = new OleDbDataAdapter("Select ID, Name from CategoriesProduct",
conn);
                bulder = new OleDbCommandBuilder(da);
                ds = new DataSet();

                DataTableMapping datam = da.TableMappings.Add("Table", "Категория
продуктов");
                datam.ColumnMappings.Add("ID", "Код");
                datam.ColumnMappings.Add("Name", "Название категории");

            }

            if (kod_table == 3)
            {
                Text = "Справочник по странам для рецептов";
                label1.Text = "Информация по странам для рецептов";

                da = new OleDbDataAdapter("Select ID, Name from Country", conn);
                bulder = new OleDbCommandBuilder(da);
                ds = new DataSet();

                DataTableMapping datam = da.TableMappings.Add("Table", "Страны");
                datam.ColumnMappings.Add("ID", "Код");
                datam.ColumnMappings.Add("Name", "Наименование");

            }

            if (kod_table == 4)
            {
                Text = "Справочник по условиям (свадьба, день рождения)";
                label1.Text = "Информация по условиям (свадьба, день рождения)";

                da = new OleDbDataAdapter("Select ID, Name from Terms", conn);
                bulder = new OleDbCommandBuilder(da);
                ds = new DataSet();

                DataTableMapping datam = da.TableMappings.Add("Table", "Условия");
                datam.ColumnMappings.Add("ID", "Код");
                datam.ColumnMappings.Add("Name", "Название");

            }

            da.Fill(ds);

            bindingSource1.DataSource = ds.Tables[0];
            dataGridView1.DataSource = bindingSource1;
            bindingNavigator1.BindingSource = bindingSource1;

            //Установка по первым двум полям
            dataGridView1.Columns[0].ReadOnly = true;
            dataGridView1.Columns[1].Width = 300;
        }

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

    //Кнопка обновить
    private void toolStripButton1_Click(object sender, EventArgs e)
    {
```

Продовження додатку Б

```
            try
            {
                da.Update(ds.Tables[0]);
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }

        }

        //Экспорт в xls
        private void toolStripButton2_Click(object sender, EventArgs e)
        {
            if (!Form1.ExportXLS(ds.Tables[0].DefaultView))
            {
                MessageBox.Show("Невозможен экспорт в Excel!");
            }
        }

        //Экспорт в html
        private void toolStripButton3_Click(object sender, EventArgs e)
        {
            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("Невозможен экспорт в HTML!");
                return;
            }

            Form4 ff = new Form4();
            ff.webBrowser1.Navigate(Application.StartupPath        +        "\\"        +
ds.Tables[0].TableName + ".htm");
            ff.ShowDialog();

        }


    }
}
```

Додаток В

Лістинг головної форми програми Form3

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

namespace Kursova1
{
    public partial class Form3 : Form
    {

        public OleDbConnection conn;
        OleDbDataAdapter da;
        OleDbCommand cmd;
        OleDbDataReader datar;
        OleDbCommandBuilder bulder;
        DataSet ds;

        public int kod_table = 1;

        //Конструктор
        public Form3()
        {
            InitializeComponent();
        }

        //Загрузка формы
        private void Form3_Load(object sender, EventArgs e)
        {
            try
            {
                // Заполнение списка SQL - запросом
                string zapros="";

                if (kod_table == 1)
                {
                    Text = "Информация о блюдах";
                    label1.Text = "Выбор категории блюд";
                    label2.Text = "Информация о категории блюд";
                    zapros = "Select ID, Name From CategoriesDish";
                }

                if (kod_table == 2)
                {
                    Text = "Информация о продуктах";
                    label1.Text = "Выбор категории продуктов";
                    label2.Text = "Информация о категории продуктов";
                    zapros = "Select ID, Name From CategoriesProduct";
                }

                cmd = new OleDbCommand(zapros, conn);
                datar = cmd.ExecuteReader();
                while (datar.Read())
```

Продовження додатку В

```
                    comboBox1.Items.Add(new          Name_List(Convert.ToInt32(datar[0]),
datar[1].ToString()));

                comboBox1.SelectedIndex = 0;
                datar.Close();
                cmd.Dispose();
                ////////////////////////////////

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

        }

        //Выбор элемента списка
        private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
        {
            try
            {
                string str = "";

                if (kod_table == 1) str = "Select ID, Name, Kol, Cena from Dishes where
";
                if (kod_table == 2) str = "Select ID, Name, Kol, Calorie, Belki, Zhiry,
Uglev from Products where ";

                str += "ID_cat = " + ((Name_List)comboBox1.SelectedItem).ID;
                da = new OleDbDataAdapter(str, conn);
                bulder = new OleDbCommandBuilder(da);
                ds = new DataSet();

                DataTableMapping datam = null;

                if (kod_table == 1)
                {
                    datam = da.TableMappings.Add("Table", "Блюда");
                    datam.ColumnMappings.Add("ID", "Код");
                    datam.ColumnMappings.Add("Name", "Название");
                    datam.ColumnMappings.Add("Kol", "Количество");
                    datam.ColumnMappings.Add("Cena", "Цена");
                }

                if (kod_table == 2)
                {
                    datam = da.TableMappings.Add("Table", "Продукты");
                    datam.ColumnMappings.Add("ID", "Код");
                    datam.ColumnMappings.Add("Name", "Наименование");
                    datam.ColumnMappings.Add("Kol", "Фасовка, г");
                    datam.ColumnMappings.Add("Calorie", "Калорийность");
                    datam.ColumnMappings.Add("Belki", "Белки");
                    datam.ColumnMappings.Add("Zhiry", "Жиры");
                    datam.ColumnMappings.Add("Uglev", "Углеводы");
                }

                da.Fill(ds);

                bindingSource1.DataSource = ds.Tables[0];
                dataGridView1.DataSource = bindingSource1;
                bindingNavigator1.BindingSource = bindingSource1;
                dataGridView1.Columns[0].Visible = false;
```

Продовження додатку В

```
                dataGridView1.Columns[1].Width = 350;
                bindingSource1.Position = 0;
            }

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

        //Обновить данные
        private void toolStripButton1_Click(object sender, EventArgs e)
        {
                try
                {
                    da.Update(ds.Tables[0]);
                }
                catch (Exception ex)
                {
                    MessageBox.Show(ex.Message);
                }
        }

        //Экспорт в Excel
        private void toolStripButton2_Click(object sender, EventArgs e)
        {
            if (!Form1.ExportXLS(ds.Tables[0].DefaultView))
            {
                MessageBox.Show("Невозможен экспорт в Excel!");
            }
        }

        //Экспорт в Html
        private void toolStripButton3_Click(object sender, EventArgs e)
        {
            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("Невозможен экспорт в HTML!");
                return;
            }

            Form4 ff = new Form4();
            ff.webBrowser1.Navigate(Application.StartupPath        +        "\\"        +
ds.Tables[0].TableName + ".htm");
            ff.ShowDialog();
        }
    }
}
```

Додаток Г

Лістинг головної форми програми 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 Kursova1
{
    public partial class Form5 : Form
    {

        public OleDbConnection conn;
        OleDbCommand cmd;
        OleDbDataReader datar;

        //Объект продуктов
        List<Name_Product> list_product = new List<Name_Product>();

        //Номер рецепта
        int ID = 0;


        public Form5()
        {
            InitializeComponent();
        }

        //Загрузка формы
        private void Form5_Load(object sender, EventArgs e)
        {
            //Заполнение combobox
            try
            {
                //Поиск рецепта
                cmd = new OleDbCommand("Select MAX(ID_Recepies) From Recepies", conn);
                    object ob = cmd.ExecuteScalar();
                    if (Convert.IsDBNull(ob)) ID = 1;
                    else ID = Convert.ToInt32(ob) + 1;
                    label6.Text = "Рецепт № " + ID;
                cmd.Dispose();

                //Заполнение стран
                string zap = "Select Id, Name From Country";
                cmd = new OleDbCommand(zap, conn);
                datar = cmd.ExecuteReader();
                while (datar.Read())
                    comboBox1.Items.Add(new          Name_List(Convert.ToInt32(datar[0]),
datar[1].ToString()));

                comboBox1.SelectedIndex = 0;

                datar.Close();
                cmd.Dispose();
```

```
                //Заполнение категорий блюд
                zap = "Select ID, Name From CategoriesDish";
                cmd = new OleDbCommand(zap, conn);

                datar = cmd.ExecuteReader();
                while (datar.Read())
                    comboBox2.Items.Add(new          Name_List(Convert.ToInt32(datar[0]),
datar[1].ToString()));

                comboBox2.SelectedIndex = 0;

                datar.Close();
                cmd.Dispose();

                //Заполнение категорий продуктов
                zap = "Select ID, Name From CategoriesProduct";
                cmd = new OleDbCommand(zap, conn);

                datar = cmd.ExecuteReader();
                while (datar.Read())
                    comboBox4.Items.Add(new          Name_List(Convert.ToInt32(datar[0]),
datar[1].ToString()));

                comboBox4.SelectedIndex = 0;

                datar.Close();
                cmd.Dispose();
            }

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

        //Отклик на категорию блюд
        private void comboBox2_SelectedIndexChanged(object sender, EventArgs e)
        {
            //Заполнение блюд
            try
            {
                string zap = "Select ID, Name from Dishes where ";
                zap += "ID_cat = " + ((Name_List)comboBox2.SelectedItem).ID;

                comboBox3.Items.Clear();

                cmd = new OleDbCommand(zap, conn);
                datar = cmd.ExecuteReader();
                while (datar.Read())
                    comboBox3.Items.Add(new          Name_List(Convert.ToInt32(datar[0]),
datar[1].ToString()));

                comboBox3.SelectedIndex = 0;

                datar.Close();
                cmd.Dispose();

            }

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

```
            }

        }

        //Отклик на категории продуктов
        private void comboBox4_SelectedIndexChanged(object sender, EventArgs e)
        {
            //Заполнение продуктов
            try
            {
                string zap = "Select ID, Name, Kol from Products where ";
                zap += "ID_cat = " + ((Name_List)comboBox4.SelectedItem).ID;

                list_product.Clear();

                cmd = new OleDbCommand(zap, conn);
                datar = cmd.ExecuteReader();
                while (datar.Read())
                    list_product.Add(new          Name_Product(Convert.ToInt32(datar[0]),
datar[1].ToString(), 0, Convert.ToInt32(datar[2])));

                datar.Close();
                cmd.Dispose();

                bindingSource1.DataSource = null;
                bindingSource1.DataSource = list_product;
                dataGridView1.DataSource = bindingSource1;
            }

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


        }

        //Формирование рецепта
        private void button1_Click(object sender, EventArgs e)
        {
            try
            {
                string zap_zag = "INSERT INTO Recepies (ID_Recepies, ID_Country, ID_Dish,
ID_Product, Kol) ";
                string str_form = "VALUES ({0}, {1}, {2}, {3}, {4})";
                string zap = "";
                int i;
                bool flag = false;
                for (i = 0; i < list_product.Count; i++)
                {
                    if (list_product[i].flag == false) continue;
                    if (list_product[i].kol <= 0 ||
                        list_product[i].kol> list_product[i].kol_t) continue;

                    zap             =             string.Format(str_form,             ID,
((Name_List)comboBox1.SelectedItem).ID,
                        ((Name_List)comboBox3.SelectedItem).ID,       list_product[i].id,
list_product[i].kol);

                    cmd = new OleDbCommand(zap_zag + zap, conn);
                    cmd.ExecuteNonQuery();
                    cmd.Dispose();
```

```
                    flag = true;
                }

                if (flag)
                {
                    MessageBox.Show("Рецепт успешно записан!", "Внимание!",
                        MessageBoxButtons.OK, MessageBoxIcon.Information);

                    ID++;
                    label6.Text = "Рецепт № " + ID;
                    comboBox1.SelectedIndex = 0;
                    comboBox2.SelectedIndex = 0;
                    comboBox4.SelectedIndex = 0;
                    dataGridView1.Refresh();
                }
            }
            catch(Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }
    }
}
```

Додаток Д

Лістинг головної форми програми 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;
using Excel = Microsoft.Office.Interop.Excel;

namespace Kursova1
{
    public partial class Form6 : Form
    {
        public OleDbConnection conn;
        OleDbCommand cmd;
        OleDbDataReader datar;
        OleDbDataAdapter da;
        DataSet ds;


        //Объект продуктов
        List<Name_Product> list_product = new List<Name_Product>();

        public Form6()
        {
            InitializeComponent();
        }

        //Загрузка формы
        private void Form6_Load(object sender, EventArgs e)
        {
            //Заполнение combobox
            try
            {

                //Заполнение номер рецептов
                string zap = "SELECT ID_recepies FROM Recepies GROUP BY ID_recepies";
                cmd = new OleDbCommand(zap, conn);
                datar = cmd.ExecuteReader();
                while (datar.Read())
                    comboBox1.Items.Add(datar[0].ToString());

                comboBox1.SelectedIndex = 0;

                datar.Close();
                cmd.Dispose();
            }

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

        //Формирование запросов и заполнение элементов управления
        private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
```

Продовження додатку Д

```
        {

            try
            {

                int ID = Convert.ToInt32(comboBox1.SelectedItem);
                bool flag = false;

                string zap = "SELECT * FROM Recepies Where ID_recepies = " + ID;

                da = new OleDbDataAdapter(zap, conn);
                ds = new DataSet();
                da.Fill(ds);


                //cmd = new OleDbCommand(zap, conn);

                list_product.Clear();

                foreach(DataRow row in ds.Tables[0].Rows)
                {
                    //Заполнение страны и блюда
                    if(flag==false)
                    {
                        cmd = new OleDbCommand("Select Name From Country WHERE
ID="+row[2], conn);
                        textBox1.Text = cmd.ExecuteScalar().ToString();
                        cmd.Dispose();
                        cmd = new OleDbCommand("Select Name From Dishes WHERE ID=" +
row[3], conn);
                        textBox2.Text = cmd.ExecuteScalar().ToString();
                        cmd.Dispose();
                        flag = true;
                    }

                    cmd = new OleDbCommand("Select Name,Kol From Products WHERE ID=" +
row[4], conn);
                    datar = cmd.ExecuteReader();
                    datar.Read();
                    list_product.Add(new            Name_Product(Convert.ToInt32(row[4]),
datar[0].ToString(), Convert.ToInt32(row[5]), Convert.ToInt32(datar[1]),true));

                    datar.Dispose();
                    cmd.Dispose();

                }

                bindingSource1.DataSource = null;
                bindingSource1.DataSource = list_product;
                dataGridView1.DataSource = bindingSource1;
            }

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

        //Экспорт в MS Excel
        private void button1_Click(object sender, EventArgs e)
        {
            try
```

Продовження додатку Д

```
{
    ////////////////////////////////////////////////////////////
    //Сформировать документ в MS Excel
    Excel.Application ExcelApp = new Excel.Application();
    ExcelApp.Visible = true;

    Excel.Workbook book = ExcelApp.Workbooks.Add();
    ExcelApp.SheetsInNewWorkbook = 1;
    Excel.Worksheet sheet = (Excel.Worksheet)ExcelApp.Sheets.get_Item(1);
    sheet.Name = "Мой лист";

    Excel.Range range, range1, startr;

    startr = sheet.get_Range("A2", "B2");
    startr.Merge(Type.Missing);
    startr.Font.Bold = true;

    startr.Font.Name = Form1.font_text.Name;
    startr.Font.Size = Form1.font_text.SizeInPoints;
    startr.Font.Color = Form1.cvet_zag;


    startr.HorizontalAlignment = Excel.XlHAlign.xlHAlignCenter;

    range1 = sheet.get_Range("A4", "B5").Cells;

    //Устанавливаем стиль и толщину линии
    range1.Borders.LineStyle = Excel.XlLineStyle.xlContinuous;
    range1.Borders.Weight = Excel.XlBorderWeight.xlThin;
    range1.HorizontalAlignment = Excel.XlHAlign.xlHAlignLeft;
    range1.VerticalAlignment = Excel.XlVAlign.xlVAlignCenter;

    range1.Font.Name = Form1.font_text.Name;
    range1.Font.Size = Form1.font_text.SizeInPoints;
    range1.Font.Color = Form1.cvet_text;

    range = sheet.Cells;
    startr.Value2      =      "Отчет      по      рецепту      №      "      +
comboBox1.SelectedItem.ToString();
    range.set_Item(4, 1, "Страна");
    range.set_Item(5, 1, "Блюдо");
    ////////////////////////////////////////////////////
    range.set_Item(4, 2, textBox1.Text);
    range.set_Item(5, 2, textBox2.Text);

    startr = sheet.get_Range("A7", "B7");
    startr.Merge(Type.Missing);
    startr.Font.Name = "Arial Black";
    startr.Font.Size = 11;
    startr.Font.Bold = true;

    startr.HorizontalAlignment = Excel.XlHAlign.xlHAlignCenter;

    //Устанавливаем стиль и толщину линии
    range1.Borders.LineStyle = Excel.XlLineStyle.xlContinuous;
    range1.Borders.Weight = Excel.XlBorderWeight.xlThin;
    range1.HorizontalAlignment = Excel.XlHAlign.xlHAlignLeft;
    range1.VerticalAlignment = Excel.XlVAlign.xlVAlignCenter;

    range1.Font.Bold = true;

    range = sheet.Cells;
```

Продовження додатку Д

```
                startr.Value2 = "Продукты";

                startr.Font.Name = Form1.font_text.Name;
                startr.Font.Size = Form1.font_text.SizeInPoints;
                startr.Font.Color = Form1.cvet_zag;


                range1 = range.get_Range("A8", "B8");

                range1.HorizontalAlignment = Excel.XlHAlign.xlHAlignCenter;
                range1.Font.Bold = true;

                int row = 8;

                Excel.Border border;
                border = range1.Borders.get_Item(Excel.XlBordersIndex.xlEdgeTop);

                //Рисование шапки
                range.set_Item(8, 1, "Наименование");
                range.set_Item(8, 2, "Количество");


                row++;

                for (int i = 0; i < list_product.Count;i++ )
                {
                     range.set_Item(row, 1, list_product[i].name);
                     range.set_Item(row, 2, list_product[i].kol);
                     row++;
                }

                range1 = range.get_Range("A8", "B" + (row-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("A8", "B8");
                range1.Font.Color = Form1.cvet_hapka;

                sheet.Columns.AutoFit();
                range1.WrapText = true;
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }
    }
}
```