**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**
**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ**

**Інститут (факультет ) автоматизації і комп'ютерних систем**
**Кафедра інформаційних технологій, штучного інтелекту і кібербезпеки**

| | |
|---|---|
| **«До захисту в ЕК»** | **«До захисту допущено»** |
| Директор інституту(декан факультету) | Завідувач кафедри |
| Андрій Форсюк | Сергій Грибков |
| (підпис) (ім'я та прізвище) | (підпис) (ім'я та прізвище) |
| «12» лютого 2024р. | «12» лютого 2024р. |

# КВАЛІФІКАЦІЙНА РОБОТА
# НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ МАГІСТРА

зі спеціальності 122 «Комп'ютерні науки»
(код і назва спеціальності)

освітньо-професійної програми Інформаційні управляючі системи та технології

на тему: Дослідження та розроблення інформаційної системи збору та обробки даних онлайн-магазину zakaz.ua

Виконав: здобувач 2 курсу, групи 3

Охріменко Владислав Володимирович _____
(прізвище, ім'я, по батькові повністю) (підпис)

Керівник Андріюк Олена Петрівна _____
(прізвище , ім'я та по батькові повністю) (підпис)

Консультанти _____ _____
(ім'я та прізвище) (підпис)

_____ _____
(ім'я та прізвище) (підпис)

_____ _____
(ім'я та прізвище) (підпис)

Рецензент Олександр Тупенко _____
(ім'я та прізвище) (підпис)

Я як здобувач(ка) Національного університету харчових технологій розумію і підтримую політику університету з академічної доброчесності. Я не надавав(-ла) і не одержував(-ла) недозволеної допомоги під час підготовки цієї роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

Здобувач _____
(підпис)

Київ — 2024р.

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ

Інститут (факультет) _ автоматизації і комп'ютерних систем _____
Кафедра __інформаційних технологій, штучного інтелекту і кібербезпеки ___
Освітній ступінь _магістр_____
**Спеціальність** __122 «Комп'ютерні науки»_____
(код і назва)
**Освітньо-професійна програма** ___Інформаційні управляючі системи_____
та технології_____
(назва)

**ЗАТВЕРДЖУЮ**
**Завідувач**
**кафедри** Інформаційних технологій,
штучного інтелекту і кібербезпеки ___
_____Грибков С.В.__
**«_19__» __грудня_____ 2023 року**

**З А В Д А Н Н Я**

**НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧА**

_____Охріменко Владислав Володимирович_____
(прізвище, ім'я, по батькові)

1. Тема роботи _Дослідження та розроблення інформаційної системи збору та обробки даних онлайн-магазину zakaz.ua_____
_____,
керівник роботи _доц. Андріюк Олена Петрівна_____,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом закладу вищої освіти від «19» грудня 2023 р. № 1006-КС_____
2. Строк подання здобувачем роботи: __23 лютого 2024_____
3. Вихідні дані до роботи: Огляд сучасного стану збору та обробки даних, включаючи загальні поняття, методи збору даних в Інтернеті та їх порівняння, огляд сучасних систем автоматизованої обробки даних у сфері електронної комерції, порівняльний аналіз розробленого проекту з існуючими аналогами на ринку_____
4. Зміст пояснювальної записки (перелік питань, які потрібно розробити):
_Дослідження методів збору даних в Інтернеті, дослідження сучасних системи автоматизованої обробки даних в сфері електронної комерції, дослідження принципів розробки програмного забезпечення, огляд розробленої системи порівнюючи з існуючими аналогами на ринку._____
_____
5. Перелік графічного матеріалу:
__Діаграми компонентів розробленого проекту, рисунки інтерфейсу розробленого проекту, інтерфейс систем автоматизованої обробки даних_____

6. Консультанти розділів роботи:

| Розділ | Прізвище, ініціали та посада консультанта | Підпис, дата | |
|---|---|---|---|
| | | завдання видав | завдання прийняв |
| 1 | Доц. Андріюк О.П. | 12.11.2023 | |
| 2 | Доц. Андріюк О.П. | 12.11.2023 | |
| 3 | Доц. Андріюк О.П. | 12.11.2023 | |
| | | | |
| | | | |
| | | | |
| | | | |

7. Дата видачі завдання: 19 грудня 2023 року

## КАЛЕНДАРНИЙ ПЛАН

| № | Назва етапів виконання кваліфікаційної роботи | Строк виконання етапів роботи | Примітка |
|---|---|---|---|
| 1 | Розгляд теоретичних основ | 12.11.2023 | Виконано |
| 2 | Дослідження сучасного стану збору та обробки даних | 22.12.2023 | Виконано |
| 3 | Дослідження принципу розробки програмного забезпечення | 29.12.2023 | Виконано |
| 4 | Розробка проекту | 02.01.2024 | Виконано |
| 5 | Оформлення роботи | 10.01.2024 | Виконано |
| 6 | Оформлення автореферату | 19.01.2024 | Виконано |
| 7 | Розробка презентації | 20.01.2024 | Виконано |
| 8 | Подання до захисту | 31.01.2024 | Виконано |
| | | | |
| | | | |
| | | | |
| | | | |

**Здобувач** _____ (підпис) Охріменко В.В. (прізвище та ініціали)

**Керівник роботи** _____ (підпис) Андріюк О.П. (прізвище та ініціали)



3

# АНОТАЦІЯ

Кваліфікаційна робота на тему «Дослідження та розроблення інформаційної системи збору та обробки даних онлайн-магазину zakaz.ua» містить 94 сторінок текстового документа, 2 додатки та 12 використаних джерел.

КЛЮЧОВІ СЛОВА: ІНФОРМАЦІЙНА СИСТЕМА, ЗБІР ДАНИХ, ОБРОБКА ДАНИХ, ZAKAZ.UA, ПРАТ «ОБОЛОНЬ», ВЕБ-ПАРСИНГ, API.

Мета роботи полягає у досліджені сучасного стану збору та обробки даних, розробка інформаційної системи автоматизованого збору та обробки даних для ПрАТ «Оболонь» з сайту zakaz.ua.

Завданням роботи є:

- Дослідити методи збору даних в Інтернеті;
- Оглянути та порівняти сучасні системи автоматизованої обробки даних;
- Дослідити принципи розробки програмного забезпечення;
- Оглянути Zakaz.UA та ПрАТ «Оболонь»;
- Огляд та розширення технічного завдання від ПрАТ «Оболонь»;
- Проектування програмного забезпечення по технічному завданню;
- Розробка програмного забезпечення;
- Написання інструкцій користувача;
- Порівняти розроблене програмне забезпечення з аналогами на ринку.

Результатами виконання завдань кваліфікаційної роботи є дослідження сучасного стану збору та аналізу даних, порівняльний аналіз методів збору даних в Інтернеті, огляд Zakaz.UA та ПрАТ «Оболонь», розроблений проек по розширеному технічному завданню, порівняння проекту з аналогами на ринку.

# ANNOTATION

The qualification work on the topic "Research and development of an information system for collecting and processing data of the online store zakaz.ua" contains 94 pages of text document, 2 appendices and 12 used sources.

KEYWORDS: INFORMATION SYSTEM, DATA COLLECTION, DATA PROCESSING, ZAKAZ.UA, OBOLON, WEB PARSING, API.

The purpose of the study is to investigate the current state of data collection and processing, to develop an information system for automated data collection and processing for Obolon from the zakaz.ua website.

The objectives of the work are:

- Investigate the methods of data collection on the Internet;
- Review and compare modern automated data processing systems;
- Explore the principles of software development;
- Review Zakaz.UA and Obolon;
- Review and expand the technical specifications from Obolon;
- Designing software according to the technical specifications;
- Software development;
- Writing user manuals;
- Compare the developed software with analogues on the market.

The results of the qualification work are a research of the current state of data collection and analysis, a comparative analysis of data collection methods on the Internet, a review of Zakaz.UA and Obolon, a project developed according to the extended technical specifications, and a comparison of the project with analogues on the market.

# Зміст

# Вступ

**Актуальність теми**

Розвиток електронної комерції в Україні постійно набирає обертів. Онлайн-магазини стають все популярнішими серед споживачів, що призводить до зростання обсягів продажів через Інтернет. У такому контексті автоматизовані системи збору та обробки даних набувають важливості, оскільки вони дозволяють ефективно управляти великими потоками інформації.

Зокрема, якщо говорити про zakaz.ua, цей онлайн-магазин має значний обсяг продажів та клієнтську базу. Розробка системи може значно полегшити процеси управління товарними запасами, моніторингу попиту, прогнозування продажів та взагалі підвищити ефективність роботи.

Дійсно, автоматизована система збору та аналізу даних може стати неоціненним інструментом для розвитку онлайн-магазину zakaz.ua. Завдяки такій системі, ви зможете ефективно контролювати товарний запас, стежити за попитом на продукти та навіть прогнозувати майбутнє зростання продажів. Одна з головних переваг автоматизованої системи полягає у швидкому та точному обробленні великих обсягів даних. Вона допоможе легко оновлювати і контролювати доступне товарне пропозицію, що необхідно для задоволення поточного попиту споживачів. Крім цього, за допомогою аналітичних інструментів такої системи, вам буде значно легше проводити ретельний аналіз продаж на регулярній основі.

**Мета дослідження**

Мета роботи полягає у досліджені сучасного стану збору та обробки даних, розробка інформаційної системи автоматизованого збору та обробки даних для ПрАТ «Оболонь» з сайту zakaz.ua.

**Завдання дослідження**

Для досягнення мети дослідження необхідно виконати наступні завдання:

- Дослідити методи збору даних в Інтернеті;
- Оглянути та порівняти сучасні системи автоматизованої обробки даних;

- Дослідити принципи розробки програмного забезпечення;
- Оглянути Zakaz.UA та ПрАТ «Оболонь»;
- Огляд та розширення технічного завдання від ПрАТ «Оболонь»;
- Проектування програмного забезпечення по технічному завданню;
- Розробка програмного забезпечення;
- Написання інструкцій користувача;
- Порівняти розроблене програмне забезпечення з аналогами на ринку.

**Визначення вимог**

Вимоги до автоматизованої системи повинні відповідати потребам ПрАТ «Оболонь» та забезпечувати її ефективну роботу, а саме:

- Збір інформації про ціни на товари в магазинах, з можливістю вибору товарів для парсингу;
- Обмеження: Парсити інформацію лише про товари, які є в наявності.
- Інтересуючи категорії товарів: Алкогольні напої (пиво, сидр, слабоалкогольні напої); напої (мінеральна вода, солодкі води та напої, енергетики).
- Необхідні дані для кожного товару: Мережа, де продається товар, код товару, повна назва товару, об’єм тари (у мл), ціна, акційна ціна (якщо є), строк дії акції (якщо є).
- Дані зберігати та накопичувати у файлі формату xlsx (Excel)

**Розробка архітектури**

Архітектура автоматизованої системи визначає структуру системи та взаємодію її компонентів.

При розробці архітектури необхідно враховувати вимоги технічного завдання.

**Розробка програмного забезпечення**

Програмне забезпечення автоматизованої системи має бути розроблено таким чином, щоб забезпечити ефективну роботу системи та відповідність її вимогам.

При розробці програмного забезпечення необхідно використовувати технології, що відповідають технічним вимогам і забезпечують виконання функціональних вимог.

**Загальна структура системи**

Система буде складатися з таких компонентів:

- Інтерфейс. Цей компонент буде використовуватися для управління системою.
- Модулі збору даних. Ці модулі будуть відповідати за збір даних.
- Модуль обробки даних. Цей модуль буде відповідати за обробку даних, отриманих від модулів збору даних.
- Модуль зберігання даних. Цей модуль буде відповідати за зберігання даних, отриманих від модуля обробки даних.

Така структура системи дозволить забезпечити її ефективну роботу та відповідність вимогам.

Технології, що будуть використовуватися

Для реалізації системи будуть використовуватися такі технології:

- Python - цю мову програмування буде використовуватися для розробки програмного забезпечення системи.
- PyQt – графічний фреймворк мови програмування Python. Використовуватиметься для розробки графічного інтерфейсу програми.
- Qt Designer – графічний інструмент для розробки інтерфейсів користувача (GUI) для програм, що використовують фреймворк Qt.
- Pandas – ця бібліотека Python використовується для аналізу та маніпуляції даними. Вона надає структури даних, такі як DataFrame, які дозволяють з легкістю читати, очищати, трансформувати та аналізувати великі обсяги даних.
- Matplotlib – це бібліотека для візуалізації даних у Python. Вона дозволяє створювати статичні, анімовані та інтерактивні візуалізації у Python.

Використовуватиметься для генерації графіків та діаграм, які допоможуть візуально представити результати аналізу даних.

- BeautifulSoup – призначена для парсингу HTML та XML документів. Вона використовується для збору даних з веб-сторінок, що дозволяє ефективно витягувати потрібну інформацію.
- Pyinstaller - цей інструмент використовується для перетворення Python програм у стандартні виконувані файли, які можна запускати на комп'ютерах без необхідності встановлення Python. Він пакує всі необхідні бібліотеки та залежності, що робить вашу програму портативною і зручною для кінцевих користувачів.
- Excel – Програма для роботи з таблицями від Microsoft, яка буде використовуватися для зберігання інформації.

**Конкретні кроки розробки**

Розробка автоматизованої системи збору та обробки даних для ПрАТ "Оболонь" може бути розділена на такі кроки:

- Визначення вимог. На цьому етапі визначаються цілі та завдання системи, а також її функціональні та нефункціональні вимоги.
- Розробка архітектури. На цьому етапі визначається структура системи та взаємодія її компонентів.
- Розробка програмного забезпечення. На цьому етапі розробляється програмне забезпечення системи.

**Наукова новизна** кваліфікаційної роботи полягає у глибокому та систематичному аналізі систем збору та обробки даних, зосереджуючись на інтеграції сучасних технологічних рішень в контексті електронної комерції і розробці програмного забезпечення згідно технічного завдання від ПрАТ «Оболонь».

**Практичне значення**

Автоматизована система збору та обробки даних для ПрАТ «Оболонь» є важливим інструментом, який може допомогти компанії підвищити свою ефективність та конкурентоспроможність.

Проект може допомогти компанії звільнити ресурси, які зараз витрачаються на ручні операції збору та обробки даних. Це дозволить компанії зосередитися на інших завданнях, більш важливих для бізнесу, таких як:

- Розвиток нових продуктів і послуг
- Розширення на нові ринки
- Покращення обслуговування клієнтів
- Отримання доступу до нових можливостей для аналізу та управління даними

Проект дозволить компанії отримувати дані в реальному часі. Це дозволить проводити більш глибокий аналіз даних і робити більш ефективні рішення.

**Особистий внесок**

У рамках дослідження та розробки автоматизованої системи збору та обробки даних я вніс наступний особистий внесок:

- Провів аналіз існуючих систем збору та обробки даних.
- Розробив архітектуру автоматизованої системи збору та обробки даних.
- Розробив програмне забезпечення автоматизованої системи збору та обробки даних.

**Апробація результатів роботи**

Апробація результатів кваліфікаційної роботи відбулася під час участі в 10-тій Міжнародній науково-технічній Internet-конференції «Сучасні методи, інформаційне, програмне та технічне забезпечення систем керування організаційно-технічними та технологічними комплексами», яка пройшла 24 листопада 2023 року.

На цій конференції була представлена доповідь, в якій були викладені основні теоретичні та практичні аспекти роботи. В ході доповіді було наголошено на значущості автоматизації процесів збору та аналізу даних. Особливу увагу було приділено програмному забезпеченню, розробленого з використанням технічного завдання від ПрАТ «Оболонь».

Тези доповіді під назвою " Дослідження та розроблення інформаційної системи збирання та опрацювання даних онлайн-магазину zakaz.ua" були опубліковані у матеріалах конференції на сторінці 210, авторами яких виступили В. В. Охріменко та М. П. Костіков. Публікація тез сприяла подальшому розповсюдженню та обговоренню отриманих результатів у наукових та професійних колах, а також визнанню значимості проведеного дослідження.

**Структура та обсяг роботи**

Кваліфікаційна робота містить вступ, 2 розділи, 3 висновка, список використаних джерел. Текст основної частини представлено на 65 сторінках у друкованому форматі. До роботи входить 16 графічних ілюстрацій, 4 таблиць та 2 додатки.

# РОЗДІЛ 1. АНАЛІЗ СУЧАСНОГО СТАНУ ЗБОРУ ТА ОБРОБКИ ДАНИХ

## 1.1. Загальні поняття та визначення

Да́ні (англ. data) — це формалізоване подання інформації, придатне для інтерпретування, пересилання чи оброблення за участю людини або автоматичними засобами. [1]

Збирання даних — процес збору та вимірювання інформації про задані змінні, встановлені систематичним способом, що дозволяють відповісти на актуальні питання й оцінити результати. Компонент збору даних певного дослідження характерний для всіх областей дослідження, серед яких фізика, суспільні науки, гуманітарні науки і бізнес. Це допомагає нам відібрати основні моменти як зібрану інформацію. У той час як методи можуть різнитися залежно від дисципліни, акцент на забезпечення точних і правдивих даних залишається. Мета всього збору даних полягає в тому, щоб відібрати якісні данні, які потім будуть переведені до аналізу усіх даних і дозволять створити переконливі та достовірні відповіді на запитання, які були поставлені.

Незалежно від області дослідження або переваги визначення даних (кількісних або якісних), точний збір даних має важливе значення для підтримки цілісності досліджень. Вибір відповідних інструментів збору даних (існуючих, модифікованих або заново розроблених) і чітке окреслення інструкцій щодо їх правильного використання, зменшують ймовірність виникнення помилок.

Формальний процес збору даних є необхідним, оскільки це гарантує те, що зібрані дані є і визначеними, і точними, і що наступні рішення, на основі аргументів, втілених у висновках, є справедливими. Процес забезпечує як базис, від якого слід вимірювати, так і в деяких випадках мітки на те, що треба покращувати.[2]

Обробка інформації — вся сукупність операцій (збирання, введення, записування, перетворення, зчитування, зберігання , реєстрація), що здійснюються за допомогою технічних і програмних засобів, включаючи обмін по каналах передачі даних.

При сучасному розвиненні програмного забезпечення існує безліч різноманітних програмних засобів обробки інформації, написаних різними мовами програмування на основі вище перелічених методів. Різноманіття ПП пов'язано із специфікою кожної галузі, в якій проводиться обробка. Наприклад, при обробці графічних зображень широко використовуються методи розпізнавання образів, криптографічні методи, що базуються на перетворенні Фур'є тощо.

Чи не кожна фірма може дозволити собі замовити у розробника ПП програму, що дозволить ефективно обробляти інформацію, пов'язану саме зі сферою діяльності цієї фірми. Такий підхід є навіть бажаним, оскільки автоматизовані системи обробки базуються на визначеній базі даних, структура якої може суттєво відрізнятися у різних підприємств, не кажучи вже про різні галузі. Одним із найрозповсюдженіших засобів обробки інформації є пакет Microsoft Officé, оскільки він встановлений майже на кожному комп'ютері. Його діапазон можливостей досить широкий, проте примітивний, якщо користувач не може безпосередньо працювати у програмному середовищі, на якому розроблений офіс. Серед засобів, доступних широкому класу споживачів є організація баз даних, відповідно виконання запитів та пошуку інформації, фільтрування інформації, графічне представлення тощо. [3]

## 1.2 Методи збору даних в Інтернеті

### 1.2.1 Веб-парсинг

Парсинг — це автоматизований збір та структурування даних з інтернету, а парсер — програма (або скрипт), яка займається цим збором за заданим алгоритмом. Об'єктами парсингу можуть бути довідники, форуми, блоги, інтернет-магазини — загалом, будь-які веб-ресурси чи їх окремі сторінки.

Здебільшого, парсинг сайтів використовують у двох цілях:

1. Технічний аналіз власного веб-ресурсу для пошуку некоректних редиректів, «битих» посилань, виявлення дублів мета-тегів, застарілої або недостовірної інформації та інших даних, важливих для SEO.

2. Парсинг з метою розвитку бізнесу. В цьому випадку парсер даних

використовується для більш швидкого закриття наступних задач:

- збір інформації з сайтів конкурентів, щоб краще дізнатись їх сильні та слабкі сторони, асортимент, особливості;
- дослідження ринку та динаміки змін (аналіз цін, попиту, пропозицій на певні товари чи послуги);
- збір відгуків та коментарів;
- наповнення нового інтернет-магазину товарними картками (наприклад, копіювання каталогу іноземного сайту та адаптація його під свій ресурс);
- створення бази лідів (парсер може знайти інформацію про те, які дії на вашому сайті виконує певна категорія клієнтів).

Також, парсинг полегшує процес переїзду сайту на новий домен. Одне із найвідповідальніших завдань технічного спеціаліста при зміні URL-адреси – перенести всі файли та бази даних так, щоб сайт працював без перебоїв. Для простих landing page використовують копіювання контенту вручну, а для багатосторінкових сайтів оптимальний варіант — парсинг. При написанні коду чи підборі вже готового сервісу враховуйте як саме потрібно перенести контент — ідентично поточній версії ресурсу (на старому домені), чи внести певні зміни (наприклад об’єднати інформацію з декількох категорій в одну) — у разі другого варіанту знадобиться більш «просунутий» та складний скрипт.

Чому парсинг кращий, ніж «ручний» збір даних?

Економія часу. Парсинг сайтів — це нескладна, але кропітка робота. Автоматизований процес збору даних знизить навантаження команди та дозволить паралельно виконувати інші задачі.

Автономність. Якщо потрібно, парсер сайтів збиратиме дані онлайн цілодобово. І зробить це швидше, ніж найспритніший співробітник.

Точність. Якісні програми чи скрипти сприймають задані параметри максимально точно та шукають лише необхідний контент — без нерелевантної та зайвої інформації. Наприклад, можна зробити конкретний запит: «ціни на троянди в 20 магазинах Шостки» — і за допомогою парсингу швидко отримати тільки потрібні результати.

Відсутність людського фактору. Людина може щось не помітити чи не надати цьому значення. У випадку з парсером це виключено, головне його правильно налаштувати.

Зручний формат даних. Можна конвертувати інформацію в потрібний формат. Наприклад з CSV в XLSX чи DOCX.

Парсинг чужих сайтів зазвичай «працює» з обох боків. Поки ви аналізуєте нових гравців на ринку, вони можуть збирати інформацію з вашого ресурсу. Парсер надає доступ до будь-якого розміщеного на сайті контенту. Найчастіше конкурентів цікавлять наступні дані:

- ціни;
- товарні картки з характеристиками;
- блог;
- всі актуальні та архівні спецпропозиції та акції;
- контакти;
- відгуки.

Таку «розвідку» проводять не лише конкуренти. Іноді, інформація потрібна журналістам для створення великих добірок з цінами або знижками брендів, чи матеріалів на іншу тему, яка потребує опрацювання великих обсягів контенту. Також, підприємці можуть моніторити декілька десятків різних компаній в якості потенційних партнерів.

Як відбувається парсинг даних?

Отримання даних парсером — це семантичний аналіз вихідного обсягу інформації. Для автоматизованого збору можна обрати один з двох форматів:

- написати скрипт самостійно. Для цього підійде майже будь-яка мова програмування (PHP, C++, Python);
- скористатися платним або безкоштовним парсером — спеціалізованою програмою для пошуку потрібної інформації у світовій мережі. (Netpeak Checker, Mozenda, Import.io та ін.).

Доступ парсера до сайту відбувається через протоколи HTTP, HTTPS, або з використанням бота з правами адміністратора. За допомогою команд задаються

межі майбутнього парсингу. Тобто, як потрібно аналізувати певний ресурс — повністю чи вибірково.

При роботі з парсером весь процес будується на введені параметрів для збору потрібної інформації та вилучення контенту. Налаштування пошуку в парсерах вводяться під конкретну задачу та мету визначення даних.

Наприклад, якщо ви продаєте кавомашини і вам потрібно дізнатись контакти з сайтів потенційних клієнтів. В програмі обирається інструмент «Парсер пошукових систем», задається ключовий запит (в даному випадку це — «кав'ярня»), кількість необхідних результатів та гео (наприклад, 50 закладів, м. Київ), а також потрібний пошуковик, мова видачі, параметр повного (всіх сторінок) або вибіркового (лише головної сторінки) аналізу веб-ресурсів. Після парсингу вся зібрана інформація відображається в сервісі у вигляді таблиці. За допомогою фільтрів можна обрати необхідні дані (тільки телефони або email) та зберегти їх в зручному форматі. (Для прикладу наведені налаштування сервісу Netpeak Checker).

Тобто, загалом парсинг можна поділити на наступні етапи:

- Постановка завдання. Користувач має чітко визначити умови відповідності вибірці. Наприклад, артикули товарів певної категорії на сайті.
- Збір релевантної інформації парсером.
- Сортування всіх даних у різні стовпці таблиці.
- Звіт з даними. В ньому можуть бути відмітки — наприклад виділення критичних помилок сайту.
- Експорт з програми в окремий документ.

Чи є законним парсинг сайтів?

Відповідь на це питання — в головному правовому джерелі нашої держави. В статті 34 «Конституції України»зазначено: «Кожен має право вільно збирати, зберігати, використовувати і поширювати інформацію усно, письмово або в інший спосіб — на свій вибір.».

В свою чергу, в статті 4 закону «Про доступ до публічної інформації» вказано про «Вільне отримання, поширення та будь-яке інше використання інформації, що

була надана або оприлюднена відповідно Закону, крім обмежень, які ним встановлено». Ознайомитись з видами обмежень можна в статті 6 Закону України.

Парсер — це програма, що шукає та аналізує контент у вільному доступі на просторах інтернету. Відповідно, така діяльність не заборонена законом. Винятком є особисті дані особи, які можуть її ідентифікувати. Парсити дозволено або деперсоніфіковані дані, або ж потрібно отримати згоду розпорядника інформації — власника сайту, на якому користувач зареєстрований. Щодо інформації, яка не є персональною — вона може вважатись конфіденційною тільки якщо про це зазначено. Так, на деяких ресурсах є розділ «Політика конфіденційності».

Крім того, згідно з Законом України «Про авторське право і суміжні права» варто враховувати можливе порушення авторських прав. Лише той, хто створив матеріал, визначає як його можна використовувати.

Який парсинг забороняється законом?

- навмисна шкода сайту (наприклад DDOS-атаки);
- пошук особистих даних користувачів, які не знаходяться у вільному доступі;
- розміщення чужого контенту від свого імені (статті, фото, відео);
- збір та-/або розповсюдження інформації, яка є комерційною або державною таємницею.[4]

Переваги:

- Автоматизований процес збору даних.
- Швидке й ефективне вилучення даних з веб-сторінок.
- Можливість збору даних з великої кількості сайтів.

Недоліки:

- Може бути складним для налаштування, якщо ви не володієте технічними навичками.
- Деякі сайти можуть блокувати веб-скрапінг.
- Не підходить для збору даних, які потребують JavaScript.

### 1.2.2. API

Application Programming Interface, що у перекладі українською —

Програмний Інтерфейс Програми. API — це набір правил і способів, якими різні програми спілкуються між собою і обмінюються даними. Ці взаємодії відбуваються з допомогою класів, методів, структур, функцій, констант однієї програми, до якої звертаються інші.

Чому ж інтерфейс? Якщо говорити простими словами, то інтерфейс – це якийсь прошарок між додатком А і додатком Б. Саме в ньому відбуваються процеси, що дозволяють цим додаткам обмінюватися інформацією та виконувати ряд функцій, пов’язаних з обома сторонами. При цьому внутрішня будова програми ховається.

Ці інтерфейси спрощують роботу користувачам та програмістами. Перші просто не замислюються, що ж стоїть за звичними функціями їхніх гаджетів, а розробникам не потрібно вивчати код інших програмістів, щоби підключити чужий продукт до свого.

Щоб стало зрозумілішим, наведемо банальний приклад: ви вирішили купити квиток у кіно, використовуючи банківську картку. Термінал звертається до API вашого банку, надсилаючи запит на оплату. Якийсь внутрішній процес, без якого складно було б пов’язати роботу двох додатків.

Навіщо потрібний API?

Можна відзначити дві основні функції API, які роблять його настільки затребуваним у розробників та користувачів.

- Інкапсуляція спрощує життя всім розробникам, оскільки окремі компоненти програми стають абстрактними. Тобто для створення нового ПЗ немає необхідності лізти в логіку низькорівневих функцій.
- На API можна непогано заробити. Наприклад, сервіси, які надають інформацію з метеовишок, стягують плату щоразу, коли здійснюється запит актуальної погоди, але тільки якщо їх API використовується у сторонніх софтах.

Чому API такі популярні у програмістів

Розробники мають особливий інтерес до API. Розглянемо 4 основні причини такої сильної тяги:

- API прискорює створення нових товарів. Простіше кажучи, програмістам не потрібно щоразу винаходити велосипед. Наприклад, можна взяти API TenserFlow та впровадити у своє ПЗ, а не самостійно витрачати час на розробку системи машинного навчання.
- Спрощує налаштування зв’язків між різними сервісами та програмами.
- Використання готових інтерфейсів – можливість заощадити не лише час, а й гроші.
- API дозволяє винести в окрему програму всю функціональність, яка має бути захищена. Це знижує можливість некоректного використання цих функцій іншими програмами. Простіше кажучи, програмний інтерфейс підвищує безпеку розробки.

Набір функцій

Опції, доступні під час роботи з API, залежать від розробників. Але загалом, лише трьома пунктами можна описати роботу інтерфейсу та методи взаємодії з ними:

- дані, які необхідно передати інтерфейсу до виконання певних функцій;
- процес, який виконує програма за допомогою API;
- дані, які ПЗ отримає на виході після обробки за допомогою API.

Можна сказати, що в результаті ми отримуємо просто приховану функцію (або набір) всередині яких відбувається деяка обробка та видача даних.

Як компанії заробляють за допомогою API

Часто компанії, діяльність яких пов’язана з розробкою складних додатків, надають клієнтам доступ до API своїх продуктів. Наприклад: розробники відеоредактора можуть брати плату за роботу з відео на своїх серверах. Прихований процес передбачає, що по API вони приймають від клієнтів файли та інструкції, а за підсумком повертають їм готовий ролик.

Важливо розуміти, що ці компанії не розкривають принципу створення своїх інтерфейсів і для розробників вони залишаються загадкою. Хоча навряд чи їхня внутрішня будова така важлива. Адже в результаті програмістам потрібен лише їхній функціонал.

Переваги:

- API може працювати некоректно. На ринку можна зустріти достатню кількість інтерфейсів, що погано працюють (або в принципі не працюють). Це з тим, що часто API створюють не вузьконаправлені фахівці.
- Оскільки API розраховано на вирішення стандартних завдань не підходить для впровадження специфічних рішень.
- Внести зміни всередині API дуже складно.
- В даний час відсутні стандарти щодо API, що веде до виникнення складнощів при взаємодії з ним.
- Вихідний код недоступний для перегляду.

Недоліки:

- Прискорює процес розробки.
- Можливість додати на сайт сторонні функції.
- Інтеграція сторонніх рішень відбувається у мінімальні терміни.
- Вразливість під час роботи з чужими компонентами знижується.
- Можливість впровадити безпечні транзакції на сайт чи додаток.
- Агрегація даних з безлічі веб-джерел.[5]

### 1.2.3 Ручний збір даних

Ручний збір даних - це метод, при якому дані збираються людиною, а не автоматизованою системою.

Ручний збір даних традиційно складався з блокнотів, ручок, дошок, секундомірів і таймерів. Вони використовувалися (і продовжують використовуватися) для запису виробничих показників, простоїв і кількості браку. Їх також використовували для вимірювання праці та руху на робочому місці.

З розвитком комп'ютерних технологій ці інструменти були замінені електронними таблицями Excel і, зрештою, окремими комп'ютерними програмами для забезпечення кращої якості даних і аналізу. Але запис і вимірювання все ще здійснювалися вручну і були схильні до людських помилок та упередженості. Самі дані були несвоєчасними, неточними і менш придатними для використання.

Ручний збір даних може здатися пристойним короткостроковим підходом,

але він може бути досить дорогим для виробників.

Мало того, що ручні стратегії є вкрай неефективними, схильними до помилок, упередженими, трудомісткими та довготривалими, вони також ускладнюють отримання точної інформації в реальному часі. Зазвичай вам доводиться чекати до кінця дня, тижня або місяця, щоб отримати зведені дані. Це означає, що ви не можете приймати своєчасні рішення, які могли б покращити ваш виробничий процес.

Ручний збір даних також є трудомістким і відволікає ресурси від виробничого процесу, спрямовуючи їх на виконання завдань, що не мають доданої вартості, - сортування, організацію та передачу даних вручну кінцевому користувачеві. Це не лише впливає на поточну продуктивність, але й заважає людям виявляти можливості та вдосконалювати процеси.

Насправді, це повністю суперечить ідеї безперервного вдосконалення, оскільки не тільки забирає час у ваших співробітників, але й перешкоджає інноваціям.

У міру того, як бізнес зростає і ви додаєте більше продуктів, ліній і співробітників, збір даних вручну стає все більш дорогим. [6]

Переваги:

- Не потребує спеціальних навичок.
- Підходить для збору даних, які не можна отримати автоматично.
- Дозволяє перевірити достовірність даних.

Недоліки:

- Трудомісткий і часомісткий процес.
- Не підходить для збору даних з великої кількості сайтів.
- Може бути схильний до людських помилок.

### 1.2.4 Інтернет-опитування

Інтернет-опитування - це метод кількісного дослідження, який використовує Інтернет для збору даних від респондентів.

Види інтернет-опитувань:

- Електронні листи: Опитування розсилаються респондентам електронною

поштою.

- Веб-сайти: Опитування розміщуються на веб-сайтах або в блогах.
- Соціальні мережі: Опитування розповсюджуються через соціальні мережі.
- Мобільні додатки: Опитування доступні через мобільні додатки.

Використання інтернет-опитувань:

Інтернет-опитування можна використовувати для різних цілей, таких як:

- Маркетингові дослідження: Вивчення думки та поведінки клієнтів.
- Соціологічні дослідження: Вивчення думки та поведінки людей з різних соціальних груп.
- Опитування громадської думки: Вивчення думки людей з актуальних питань.
- Освітні дослідження: Вивчення думки та поведінки учнів.

Переваги:

- Швидкість
- Економія коштів
- Зручність
- Широкий охват
- Анонімність

Недоліки:

- Не всі люди мають доступ до Інтернету.
- Можливість самостійного відбору респондентів може призвести до нерепрезентативних результатів.
- Деякі респонденти можуть не відповідати на питання чесно або уважно.

### 1.2.5 Порівняння методів збору даних в інтернеті

Кожен з методів збору даних має свої переваги та недоліки. Вибір методу залежить від потреб, бюджету та навичок. Результати порівняння ви можете перегляну на таблиці 1.1.

Веб-парсинг - хороший вибір, якщо вам потрібно зібрати велику кількість даних з веб-сайтів.

API - хороший вибір, якщо вам потрібно додати на сайт сторонні функції або

інтегрувати його з іншими системами.

Ручний збір даних - хороший вибір, якщо вам потрібно зібрати невелику кількість даних або якщо вам потрібна висока точність.

Інтернет-опитування - хороший вибір, якщо вам потрібно швидко та економно провести опитування великої кількості людей.

Таблиця 1.1. Порівняння методів збору даних в Інтернеті

| Метод | Переваги | Недоліки |
|---|---|---|
| Веб-парсинг | - Автоматизований процес збору даних.<br>- Швидке й ефективне вилучення даних з веб-сторінок.<br>- Можливість збору даних з великої кількості сайтів. | - Може бути складним для налаштування, якщо ви не володієте технічними навичками. - Деякі сайти можуть блокувати веб-скрапінг. - Не підходить для збору даних, які потребують JavaScript. |
| API | - Прискорює процес розробки.<br>- Можливість додати на сайт сторонні функції.<br>- Інтеграція сторонніх рішень відбувається у мінімальні терміни.<br>- Вразливість під час роботи з чужими компонентами знижується.<br>- Можливість впровадити безпечні транзакції на сайт чи додаток.<br>- Агрегація даних з безлічі веб-джерел. | - API може працювати некоректно.<br>- Оскільки API розраховано на вирішення стандартних завдань не підходить для впровадження специфічних рішень.<br>- Внести зміни всередині API дуже складно.<br>- В даний час відсутні стандарти щодо API, що веде до виникнення складнощів при взаємодії з ним.<br>- Вихідний код недоступний для перегляду. |
| Ручний збір даних | - Не потребує спеціальних навичок.<br>- Підходить для збору даних, які не можна отримати автоматично.<br>- Дозволяє перевірити достовірність даних. | - Трудомісткий і часомісткий процес.<br>- Не підходить для збору даних з великої кількості сайтів.<br>- Може бути схильний до людських помилок. |
| Інтернет-опитування | - Швидкість<br>- Економія коштів<br>- Зручність<br>- Широкий охват<br>- Анонімність | - Не всі люди мають доступ до Інтернету.<br>- Можливість самостійного відбору респондентів може призвести до нерепрезентативних результатів.<br>- Деякі респонденти можуть не відповідати на питання чесно або уважно. |

## 1.3 Огляд сучасних систем автоматизованої обробки даних у сфері електронної комерції

Системи автоматизованої обробки даних відіграють важливу роль у сфері електронної комерції, допомагаючи компаніям автоматизувати завдання, пов’язані з обробкою замовлень, обслуговуванням клієнтів і маркетингом.

### 1.3.1 Системи планування ресурсів підприємства

ERP-система (enterprise resource planning) – це програмне забезпечення для планування ресурсів підприємства, що допомагає контролювати внутрішні процеси та приймати важливі рішення щодо розвитку бізнесу в режимі реального часу.

Передумови створення ERP-систем

Вперше такого роду системи з’явилися на початку 90-х років минулого століття, як реакція ринку на ускладнення внутрішніх процесів та необхідність у збалансованому та комплексному керуванні всіма ресурсами компанії. Першими їх почали застосовувати промислові підприємства і вже незабаром, довівши свою ефективність, ERP-системи стали активно застосовуватись у сфері послуг, державних структурах та некомерційних організаціях.

Минуло більше 30 років, системи ERP стали ключовим інструментом менеджменту і продовжують активно розвиватися та використовуватися десятками тисяч компаній у різних сферах.

«Обсяг світового ринку програмного забезпечення ERP на рік становить понад 25 мільярдів доларів США і зростає на 10-20% щороку».

Як працює ERP?

Загалом система ERP – це набір окремих модулів, які пов’язані та інтегровані між собою. Кожен з цих модулів відповідальний за конкретну сферу бізнесу: фінанси, виробництво, кадровий облік, маркетинг та продажі, планування, закупівлі, логістика та інші – детальніше про користь окремих з них розповімо нижче.

ERP, подібно до центральної нервової системі, збирає сигнали та інформацію з усіх ділянок організму, тобто підприємства. Ба більше, ERP-система, як і ЦНС, розвивається, а функціональність її модулів перманентно зростає.

Важливість та актуальність впровадження ERP

Основний принцип ERP – централізований збір інформації. Всі дані знаходяться в єдиному сховищі, що дає можливість всім користувачам – від генерального директора до працівників бухгалтерії – зберігати, створювати та використовувати єдині актуальні дані підприємства. Тобто, якщо зараз компанія користується окремими базами даних для моніторингу складської звітності, обробки замовлень та бухгалтерського обліку – використовуючи при цьому велику кількість ненадійних електронних таблиць, то ERP об'єднує ці процеси в єдиний та зрозумілий потік інформації.

В умовах сучасного і динамічного ринку наявність гнучкого та безпечного програмного забезпечення, яке дозволяє приймати швидкі рішення на основі детальної аналітики, є вкрай необхідним.

Важливо також розуміти: якщо ваш бізнес націлений на міжнародні практики ведення бізнесу та зручну взаємодію з гравцями прогресивного ринку – відмова від застарілих та ненадійних систем є головним кроком до безпеки та розумного масштабування вашого бізнесу.

Основні переваги для бізнесу

Цілісна картина та глибоке розуміння поточної ситуації: люди, котрі приймають рішення, мають єдине і надійне джерело інформації, що надходить з усіх «артерій» вашого підприємства. Аналітика, створена на базі штучного інтелекту, дозволить швидко отримати відповіді на критичні питання, вчасно розпізнати тривожні сигнали та розглянути кожну ділянку вашого бізнесу з максимальною деталізацією.

Збільшення продуктивності: автоматизація бізнес-процесів дозволяє використати заощаджений час ваших працівників на розв'язання інших завдань.

Краща взаємодія з клієнтами: Централізоване сховище даних надає всю необхідну інформацію про клієнтську базу, що спрощує комунікацію і зменшує затрати часу на обслуговування клієнтів. А точно зібрані та проаналізовані дані про взаємодію з клієнтами допоможуть вам оптимізувати стратегію і бути більш точними у прогнозуванні попиту.

Безпека даних: Велика кількість окремих систем для обробки інформації несе за собою більші ризики. Єдиний та надійний центр обробки даних застрахує персональні дані ваших клієнтів та працівників, а також фінансові дані, від небажаних загроз.

Користь для окремих складових бізнесу

Система ERP допомагає зруйнувати бар'єри між окремими ділянками підприємства та організувати єдиний центр управління та планування. Нижче коротко розказано про те, які модулі містить в собі Microsoft Dynamics 365 Business Central.

- Управління фінансами (Finance management)

ERP автоматизує процес фінансового обліку згідно з міжнародними та національними стандартами. У системі здійснюється планування бюджетів, контроль їхнього виконання, а також побудова регламентованої та фінансової звітності.

- Продажі та маркетинг (Sales and marketing)

ERP дозволяє вибудувати процес продажів, починаючи з управління цінами та товарами, закінчуючи формуванням документів продажу та відвантаження зі складу. У модулі маркетингу можливо вибудовувати персоналізовані customer journey, базуючись на накопиченій інформації про клієнта та на підказках штучного інтелекту.

- Закупівлі та кредиторська заборгованість (Purchasing and payables)

Що стосується процесу закупівель, важливим акцентом є те, що система дозволяє виконувати планування постачання товарів згідно з потребами виробництва та попитом споживачів.

- Планування постачання (Supply planning and availability)

ERP-система пропонує функціональність для планування маршрутів та термінів постачання, виходячи з аналізу потреб виробництва, власного споживання та прогнозів продажу.

- Управління проєктами (Project management)

Планування та облік виконання проєктів, планування ресурсів на проєкт.

- Управління сервісом (Service management)

Управління заявками на обслуговування, регламентне обслуговування, організація роботи польових співробітників.

- Управління складом (Warehouse management)

Модуль для оптимізації та автоматизації управління складом. Складний ланцюжок руху товарів на складі дробиться на дрібніші операції та завдання, котрі виконують конкретні користувачі з контролем виконання цих операцій.

- Виробництво (Manufacturing)

Основні завдання ERP у процесі виробництва – розумне планування ресурсів задля забезпечення поточного попиту, контроль процесу виробництва, управління якістю, оптимізація виробничих витрат тощо. А також формування та калькуляція наскрізної собівартості продукту чи послуги. [7]


*Рис.1.1 – Функції Microsoft Dynamics 365 Business Central*

### 1.3.2 Система управління взаємовідносин з клієнтами

CRM-система (customer relationship management) — це спосіб

управління взаємовідносинами з клієнтами та оптимізації бізнес-процесів. CRM допомагає організувати роботу з потенційним клієнтом, відстежувати дії клієнтів та автоматизувати комунікації.

Переваги використання CRM-системи

CRM-система збирає дані кожного клієнта в одній панелі управління. За допомогою цієї інформації ваша команда може відстежувати шлях покупця та робити релевантні пропозиції на кожному етапі. В результаті цикл продажів скоротиться на 8-14%.

CRM допомагає визначити інтереси та переваги ваших клієнтів. Це дозволяє надавати персоналізований досвід та створювати релевантні маркетингові кампанії. Згідно з Findstack, 74% респондентів стверджують, що CRM дають їм кращий доступ до даних клієнтів, дозволяючи надавати більш індивідуальний підхід до обслуговування.

Завдання CRM-системи

Існує безліч сервісів для малих, середніх та великих компаній із різних сфер бізнесу. Проте є низка основних завдань, які ви можете побачити нижче.

- Консолідація даних клієнтів. CRM-система повинна збирати контакти ваших лідів та покупців, їх демографічні дані та іншу інформацію, забезпечуючи легкий доступ до неї.
- Відстеження взаємодій та активності. CRM-системи дозволяють відстежувати комунікацію з клієнтами в чатах через менеджери, телефон, email та інші канали.
- Оцінка продуктивності. Хороша CRM-система дозволяє отримувати звіти з детальними даними щодо ефективності взаємодії компанії з клієнтами.
- Автоматизація рутинних процесів. Автоматизація маркетингу та продажу – це основа будь-якої CRM-системи.

CRM-система може вирішувати різні завдання, із якими стикаються компанії. Давайте розглянемо основні види програмного забезпечення управління взаємовідносинами з клієнтами.

Види CRM-систем

Універсальна CRM-система – це рідкість. Зазвичай одне програмне забезпечення виконує певне завдання краще, ніж інші. Залежно від своїх можливостей, CRM можна поділити на декілька видів.

- Операційні системи CRM. Допомагають виконувати повсякденні процеси вашої компанії та автоматизувати рутинні завдання.
- Аналітичні системи CRM. Це величезні бази даних з детальною інформацією про ваших клієнтів та бізнес-процеси.
- Колективні системи CRM. Допомагають підвищити ефективність взаємодії між різними відділами компанії.

Кожне бізнес-завдання потребує конкретного рішення. Те, що працює для досягнення маркетингових цілей, може не підходити для обслуговування клієнтів. Тому вам необхідно серйозно поставитися до вибору платформи, яка зможе задовольнити всі ваші бажання та вирішити нагальні питання.

SendPulse

SendPulse – відмінний вибір для тих, хто шукає інструмент із ефективним набором функцій автоматизації маркетингу. Для залучення лідів ви можете створювати професійні мультиканальні форми підписки, попапи та чат-боти у соціальних мережах. Конструюйте автоматизовані ланцюжки повідомлень для вирощування лідів та своєчасну реакцію на їх повідомлення. Також сервіс допомагає створювати сценарії повідомлень чат-ботів для комунікації з користувачами в Telegram, Facebook Messenger, WhatsApp, Viber та Instagram.

CRM-система від SendPulse відмінно підійде тим, хто ще не пробував використовувати CRM і веде облік продажів на папері. Ви можете почати використовувати нашу CRM безплатно для 5 користувачів в акаунті. Інтуїтивний інтерфейс в єдиному стилі з іншими можливостями компанії допоможе легко розібратися в системі, якщо ви раніше не використовували SendPulse.

У CRM ви можете зберігати та оновлювати інформацію про клієнтів, створювати угоди, додавати потрібні вам статуси, щоб візуалізувати процеси на канбан-дошці.

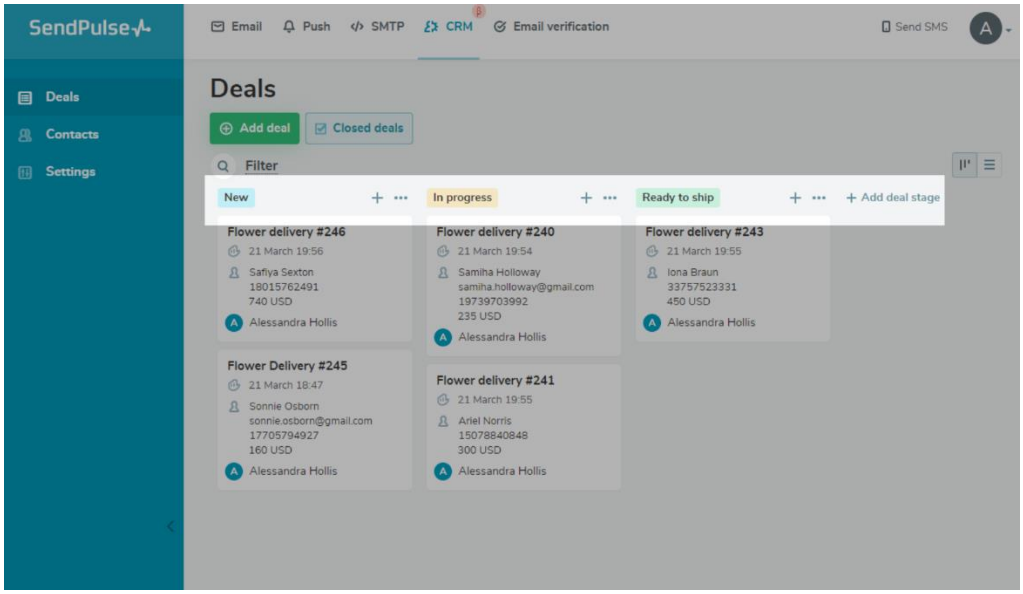



*Рис.1.2 – Вигляд функціоналу SendPulse*

Натиснувши на картку угоди, ви отримаєте повну інформацію по ній: дані про клієнта, про угоду (коли створена, вручну або з автоматичного джерела, історію її переміщення по етапах та фахівця відповідального за її укладання), коментарі. Ви можете редагувати та додавати потрібні поля, залишати коментарі, зв'язуватись з клієнтом через e-mail або чат-бот, якщо він був ініціатором спілкування через цей канал. Теги допоможуть швидко знайти потрібний контакт та сегментувати клієнтів.

Якщо ви надсилаєте email розсилки через SendPulse та використовуєте чат-ботів, ви можете робити автоматичні розсилки прямо з CRM по тригеру. Це може бути створення угоди, тобто, коли користувач натискає кнопку «Купити» у чаті з вашим ботом, або перехід угоди на новий етап, наприклад, «Чекає на оплату» або «Готовий до відправки».

Ви також можете додавати членів своєї команди до CRM і призначати відповідальних за кожну угоду. Колеги, яких ви додасте до сервісу, зможуть бачити та працювати з усіма угодами, але ви можете обмежити їх доступ до інших сервісів SendPulse.

CRM від SendPulse є абсолютно безкоштовною. Ви можете вільно зберігати всю інформацію, створювати ефективну воронку продажу, щоб працювати з клієнтами за певним сценарієм, користуватися різними каналами для спілкування та закривати угоди.

KeepinCRM

Відмінне рішення для малого та середнього бізнесу. Сервіс дозволяє вести комунікацію з клієнтами, керувати продажами, підрядниками та складами, вести фінансовий облік та документообіг.

KeepinCRM надає можливість інтеграції з Новою Поштою та УкрПоштою. Сервіс дозволяє автоматично заповнювати дані про клієнтів та товари, формувати та друкувати ТТН, відстежувати статуси доставки, формувати списки кур'єра. У CRM-системі щомісяця проводять оновлення та додають нові функції для підвищення ефективності роботи.

KeepinCRM забезпечує контроль залишків на складах, надає можливість передавати товари між філіями, налаштовувати автоматичне повернення товарів на склад, робити розсилку повідомлень клієнтам по SMS та багато іншого. CRM-система інтегрується з Instagram та Facebook Leads, Rozetka, Prom, WordPress, TurboSMS та іншими сервісами. На даний момент компанія вже працює над створенням інтеграцій із Telegram, Viber, ПриватБанк, Monobank, Liqpay, Google Sheets, MailChimp та SendPulse.

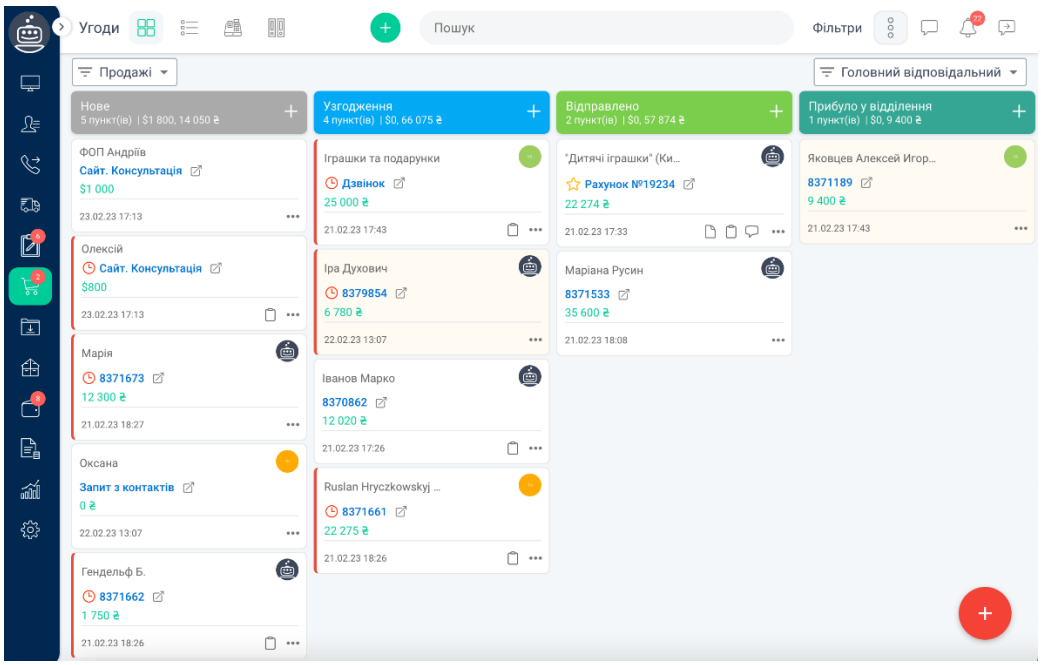



*Рис.1.3 – Вигляд функціоналу KeepinCRM*

Безкоштовний тариф доступний для одного користувача на мінімально необхідний функціонал. Платний тариф коштує 299 гривень на місяць і надає вам

додаткового користувача та повний функціонал, без обмежень.

KeyCRM

KeyCRM підійде різним видам бізнесу. Сервіс допоможе вам здійснювати продажі онлайн на маркетплейсах, в Instagram та в інтернет-магазинах. Платформа ідеально підійде також тим, хто займається бізнесом у сфері послуг. Сміливо обирайте KeyCRM якщо ви займаєтеся навчанням, консалтингом, B2B або B2C. За допомогою цієї платформи ви зможете збирати лідів, заявки та замовлення з різних каналів, таких як сайти, маркетплейси, месенджери та Instagram.

Сервіс дозволяє приймати замовлення з усіх маркетплейсів і зберігати в одній системі. Є вбудовані модулі Etsy, eBay, Amazon, Rozetka, Prom, Allo та Shopify. Також перевагою є те, що сервіс швидко оформлює ТТН і автоматично відправляє дані на маркетплейси.

Якщо ж ви займаєтеся продажами в Insagram, ви точно повинні оцінити функцію збирання всіх повідомлень з Instagram Direct, коментарів і реакцій до постів та сторіс в CRM. Таким чином ви можете контролювати всі діалоги з користувачами і правильно розподіляти завдання серед членів вашої команди. KeyCRM піклується про підрахунок складу товарів та облік залишків. Тож ви будете знати, коли товари закінчуються.

Для тих, хто має інтернет магазин, сервіс гарантує підключення будь-якого сайту: WooCommerce, Opencart, Prestashop. CRM має величезний список інтеграцій, тому ви зможете відразу підключити необхідні сервіси. KeyCRM дозволяє побачити актуальний статус замовлень та залишки товарів в одному вікні.

Розробники платформи попіклувалися і про людей, які мають бізнес у сфері послуг. За допомогою цієї CRM ви можете збирати заявки в одному вікні, користуватися календарем записів на послуги, збирати запити з Telegram, Viber, Instagram, Facebook, форм на сайті та вести облік витрат по матеріалам.

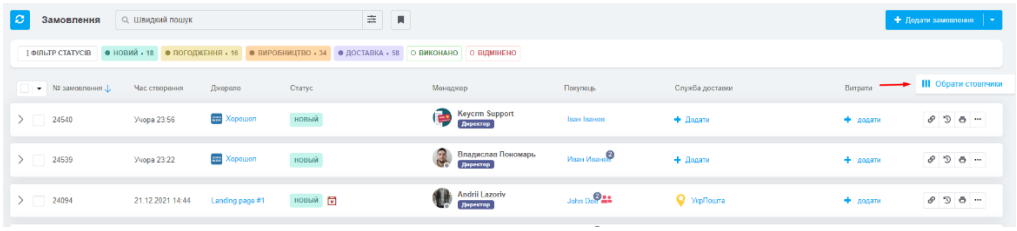


*Рис.1.4 – Вигляд функціоналу KeepinCRM*

Платформа має 30-денний пробний період. Для тривалого використання рекомендуємо придбати тарифний план, ціна якого буде коливатися залежно від кількості замовлень, заявок та повідомлень. Базова оплата становить $19 в місяць за 200 замовлень, 2000 заявок, 20000 повідомлень з чатів (Instagram/Viber/Telegram).

NetHunt

NetHunt це ідеальне рішення як для малого так і для великого бізнесу. Сервіс допомагає B2B та B2C компаніям автоматизувати бізнес-процеси, задачі та контроль за виконанням завдань командою. За допомогою платформи ви можете відповідати на повідомлення клієнтів з різних месенджерів в одному вікні. Нові ж чати автоматично додаються до вашої CRM та до картки клієнта. Всі дані клієнтів та чати з ними будуть надійно збережені в системі. Більш того, ви можете структурувати всю інформацію як вам зручно.

Якщо ви хочете зосередитися на генерації лідів, CRM система прийде вам на поміч. Платформа допоможе зібрати нових лідів, налаштувати сповіщення про них і підігріти потенційних клієнтів шляхом відправлення автоматичних листів.

З NetHunt ви зможете звільнити своїх співробітників від рутинних завдань і дати їм можливість зосередитися на більш серйозних проблемах клієнтів. Платформа подбає про збір та прогрів лідів, введення даних і продажі.

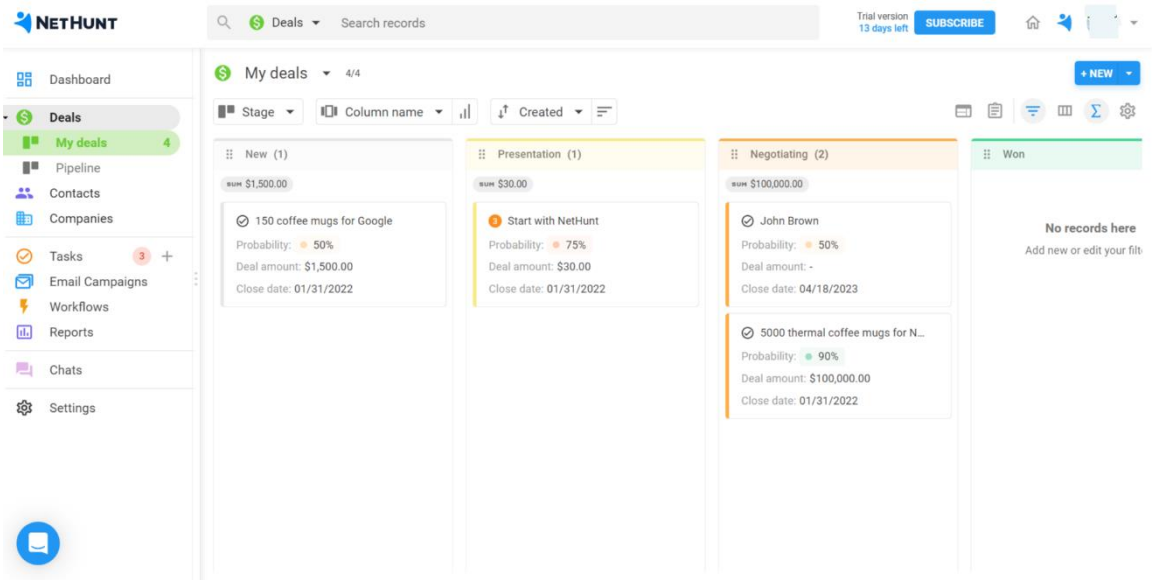



*Рис.1.5 – Вигляд функціоналу NetHunt*

Ціни на тарифні плани починаються з $30 за місяць. Сервіс дає можливість спробувати користуватися CRM безкоштовно протягом 14 днів.

Zendesk Sell

Zendesk Sell – це модель операційної CRM-системи, яка дозволяє телефонувати клієнтам, надсилати їм листи та планувати зустрічі в рамках одного сервісу. Ви можете автоматизувати обмін даними прямо всередині системи або за допомогою зовнішніх сервісів, таких як Mailchimp, Pandadoc та ваші власні програми, створені за допомогою платформи Zendesk Apps.

Цей інструмент дає чудові можливості для керування лідами, починаючи від пошуку клієнтів по кількох каналах і закінчуючи створенням їх точних профілів. Більше того, Zendesk Sell має вбудовану базу даних із більш ніж 20 мільйонами компаній та 200 мільйонами професіоналів, тому ви можете знаходити потенційних клієнтів на основі створених профілів покупців.

Ще однією важливою перевагою Zendesk Sell є його глибока аналітика, яка включає більше 30 готових до використання звітів та інформаційних панелей, що налаштовуються. Останні дозволяють вам відстежувати активність продажів, кількість дзвінків, тривалість укладання угод, результативність та багато іншого.

Нижче ви бачите розділ “Звіти”. У ньому відображаються найважливіші показники, які можна порівнювати зі своїми фінансовими цілями реального часу.

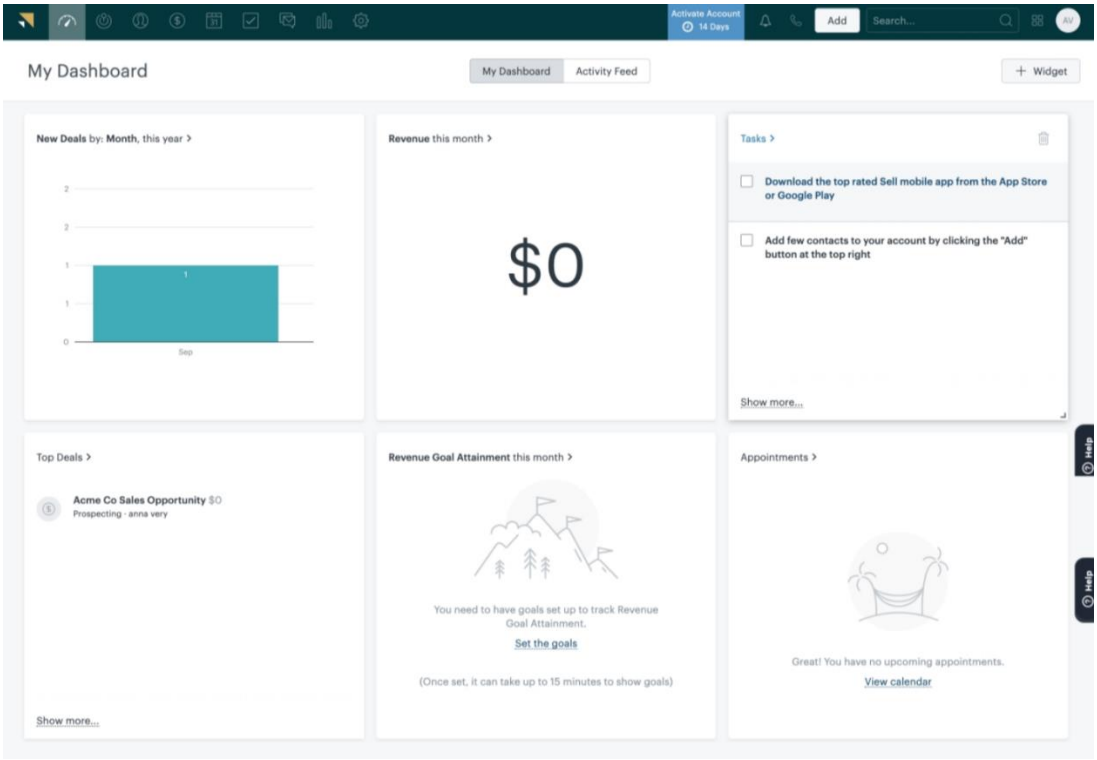



*Рис.1.6 – Вигляд функціоналу Zendesk Sell*

Безкоштовного тарифу немає, є пробна версія на 14 днів. Платний тариф коштує від $25 на місяць за одного користувача.

Pipedrive

Pipedrive — це візуальна, проста у використанні операційна CRM-система, яка легко налаштовується. Як і Zendesk Sell, Pipedrive дозволяє дзвонити, надсилати листи та планувати зустрічі прямо з панелі управління. Ви також можете зберігати документи для швидкого доступу.

CRM-система надає звіти, що налаштовуються, і велику кількість інтеграцій, включаючи власні мобільні додатки, сервіси Google і Microsoft, а також більше 150 інструментів. Pipedrive дозволяє створити чат-бота для свого сайту або підключити чат в реальному часі.

У сервісі є віртуальний консультант, створений на базі штучного інтелекту, який рекомендує способи автоматизації процесів підвищення ефективності роботи. Більше того, на підставі вашої поведінки він дає підказки про те, як підвищити результативність продажів.

Pipedrive дозволяє налаштовувати воронки відповідно до вашого циклу

продажів. На скріншоті показаний спосіб організації такого процесу на підставі взаємодії відділу продажу з потенційним клієнтом.

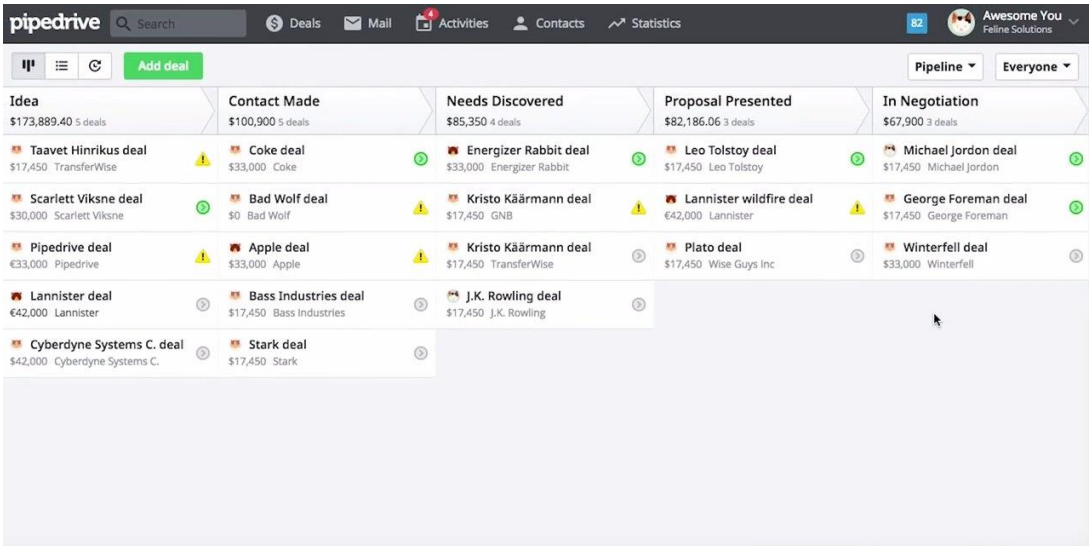



*Рис.1.7 – Вигляд функціоналу Pipedrive*

Безкоштовного тарифу немає, є 14 днів пробного періоду. Платний тариф коштує від $15 на місяць за одного користувача.

Zoho CRM

Програмне забезпечення Zoho надає користувачам багатоканальний зв'язок. Це дозволяє вам автоматизувати процес зв'язку з клієнтами за допомогою телефону, електронної пошти, чату та соціальних мереж. Якщо клієнти зв'яжуться з вашою компанією, ви одразу будете сповіщені. В результаті ви отримаєте звіт про ефективність комунікації з потенційними клієнтами та покупцями. За допомогою розмовного помічника ви можете швидко знайти необхідну інформацію у своїй CRM і отримати прогнози щодо угод з певними клієнтами. Таким чином ви зможете визначити потенційних клієнтів, які з більшою ймовірністю принесуть вам прибуток. Крім того, помічник буде нагадувати про завдання та вносити пропозиції.

Сервіс допоможе створити інтерфейс користувача CRM відповідно до ваших уподобань. Ви можете персоналізувати його так, щоб програмне забезпечення відповідало вашим вимогам і потребам. Інформаційна панель проста і зручна. Це дозволяє користувачам тримати під контролем завдання з продажу та

повідомлення.

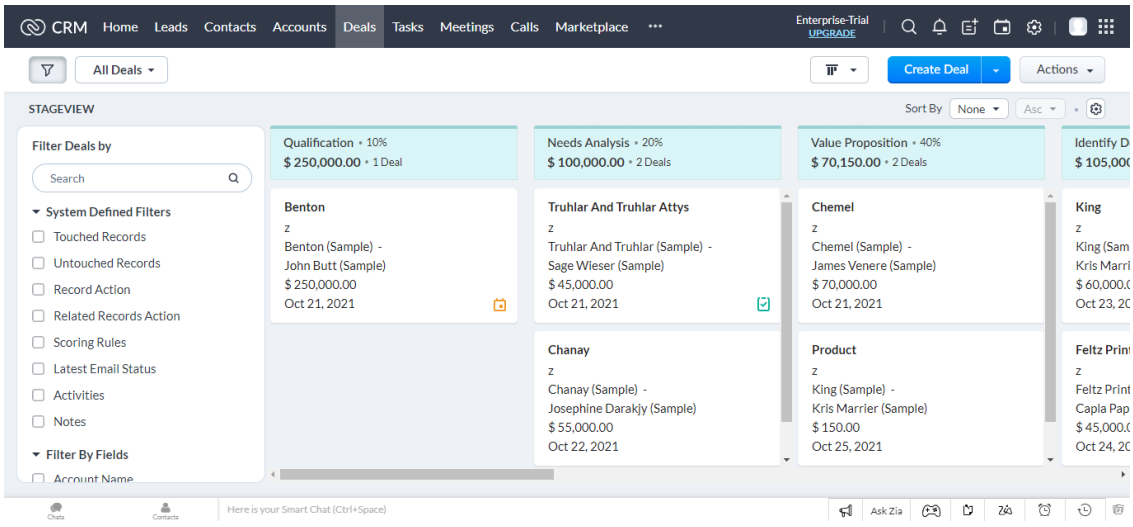



*Рис.1.8 – Вигляд функціоналу Zoho CRM*

Zoho надає користувачам безкоштовну 15-денну пробну версію з можливістю додавати три користувача. Є 4 платні плани, вартість яких починається від €20 за місяць за умови щорічної оплати. Найдешевший план включає робочі процеси, правила оцінки, розсилку та спеціальні інформаційні панелі.

monday.com

CRM – ідеальне рішення для тих, хто хоче зробити кожну угоду видимою та покращити командну роботу. Єдина платформа — це все, що вам потрібно для зберігання важливих документів, контактів і завдань. Програмне забезпечення доступне як і в мобільній так і в версії для ПК, тож ви можете працювати будь-коли. Крім того, платформа дозволяє використовувати офлайн-рішення для керування компанією навіть без підключення до інтернету.

Ви можете використовувати готові шаблони для створення робочих процесів. Вони корисні як для малих, так і для великих команд для створення, делегування завдань і керування угодами. Функція відстеження клієнтів дає змогу отримати уявлення про взаємодію з клієнтами та дії, які потрібно виконати далі.

Інтуїтивно зрозумілий інтерфейс допомагає візуалізувати ваші продажі та відстежувати потенційних клієнтів. Якщо у вас є досвід роботи з платформами, заснованими на канбан-дошці, тоді вам буде легко налаштувати та використовувати CRM. Ви можете коригувати інформаційну панель відповідно до

своїх уподобань і потреб.

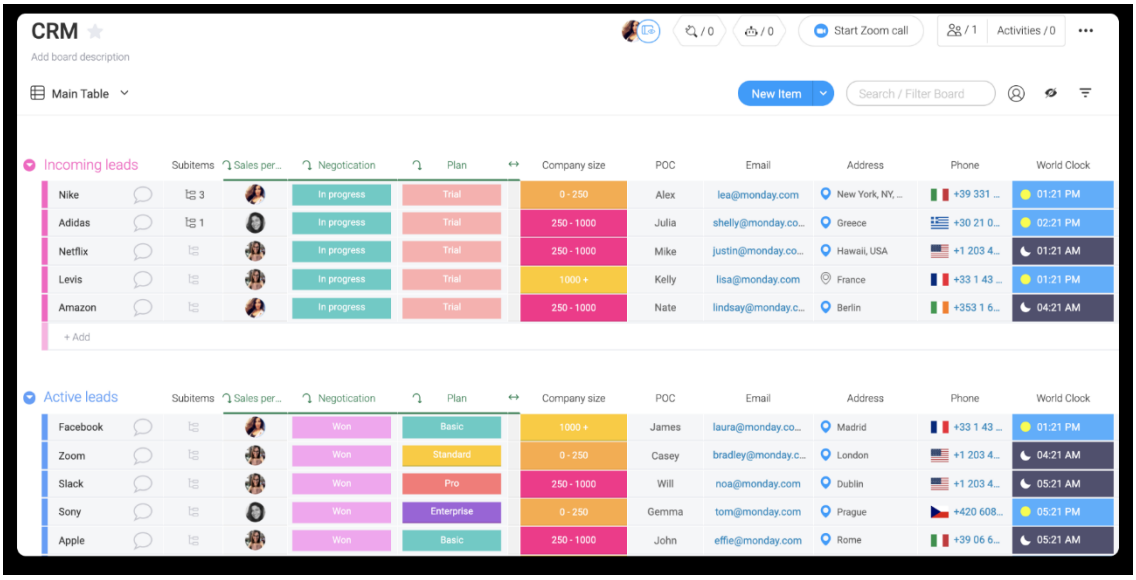



*Рис.1.9 – Вигляд функціоналу monday.com*

CRM від monday.com пропонує користувачам спробувати його безкоштовний тарифі функції. Він охоплює два користувача, необмежену кількість документів і понад 200 шаблонів. Але якщо вам потрібен доступ до таких функцій, як керування контактами, візуальні канали продажів, мобільна CRM, користувацькі поля та необмежена кількість контактів, радимо придбати платний тариф, який починається від €10 за користувача на місяць.

Freshsales

Платформа допомагає членам вашої команди заощадити час на впорядкуванні інформації про клієнтів і зосередитися на більш важливих завданнях, таких як продажі. Freshworks прагне залучити потенційних клієнтів і зафіксувати враження клієнтів від бренду. У результаті ви можете підвищити ефективність і розробити правильні маркетингові стратегії.

Freddy Al — ще один корисний інструмент, який дає змогу переглядати прогноз продажів для кожного клієнта на основі купівельної поведінки. Прогностична оцінка контактів допомагає зрозуміти, які потенційні клієнти з більшою ймовірністю стануть вашими справжніми покупцями.

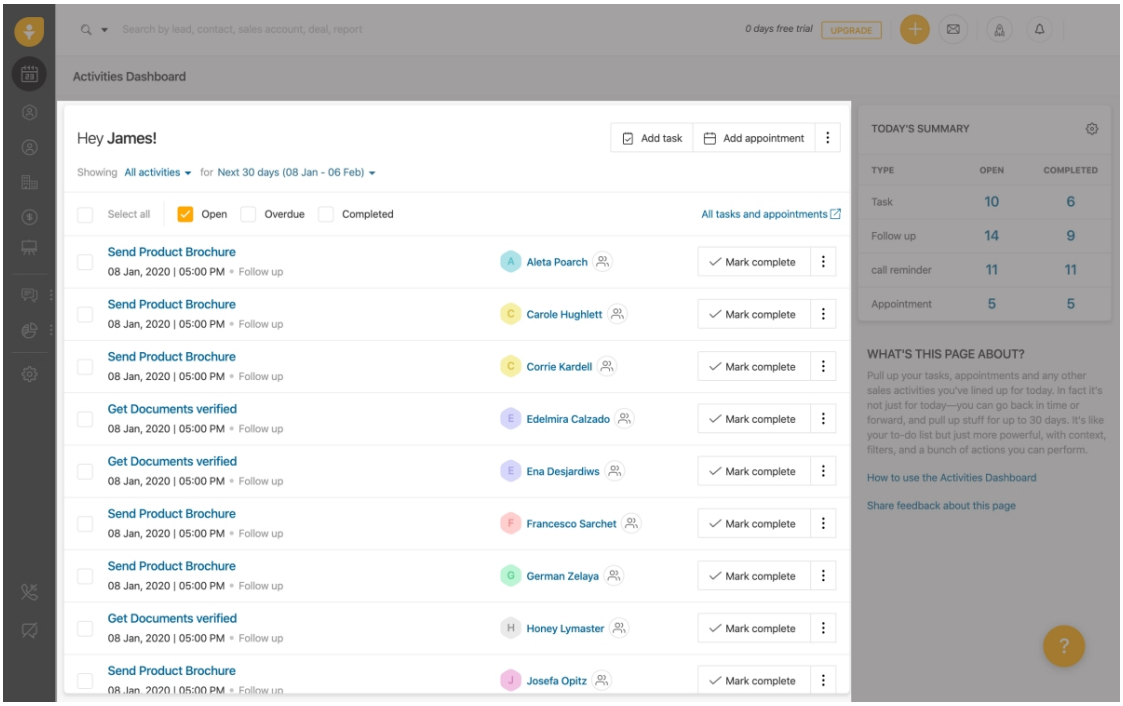



*Рис.1.10 – Вигляд функціоналу Freshsales*

Перше, що варто згадати, це те, що компанія має 21-денний пробний період. Це дозволяє користувачам тестувати програмне забезпечення без заповнення даних кредитної картки. Якщо ви вже вирішили, що Freshworks CRM вам ідеально підходить, то можете вибрати один із трьох їх платних планів. Чим дорожча підписка, тим більше контактів, робочих процесів і додаткових можливостей. Найдешевший план коштує $18 на місяць. Ви матимете доступ до сегментації, управління угодами, відстеження відвідувачів, тощо.

Trello

Це гнучкий і простий у використанні онлайн-інструмент. У сервісі ви звернете увагу на дошки, списки, картки. Списки визначають різні етапи проекту, тоді як картки представляють роботу, яку вам потрібно виконати під час проекту. Цей інструмент дає вам можливість інтегрувати всі необхідні додатки. Ви можете підключити свою дошку до програм, на які покладається ваша команда, наприклад Dropbox, Gmail, Outlook, Jira та багато інших.

Інструментом можуть користуватися як окремі особи, так і команди. За допомогою Trello члени команди можуть працювати над проектом, не впливаючи на оновлення.

Ви можете побачити список завдань та їхні статуси. Система дає можливість відслідковувати, на якому етапі перебуває кожне завдання та чи буде воно успішно завершено перед дедлайном.

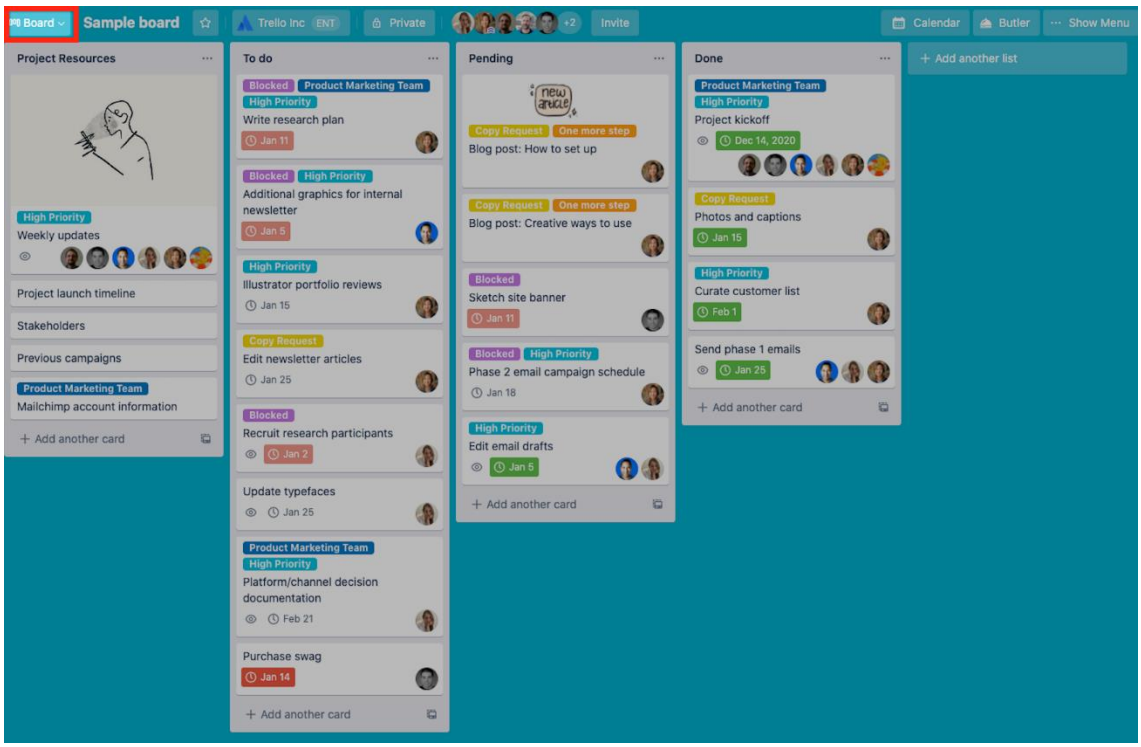



*Рис.1.11 – Вигляд функціоналу Trello*

Сервіс має безкоштовний план, який ви можете використовувати на постійній основі. Платний тариф обійдеться вам в $12.50 на місяць за одного користувача з доступом до всіх необхідних інструментів. [8]

### 1.3.3 Системи управління складом

Система управління складом (WMS) - це програмне забезпечення та процеси, які дозволяють організаціям контролювати та адмініструвати складські операції з того моменту, коли товари або матеріали надходять на склад до їх вибуття. Операції на складі включають управління запасами, процеси відбору та аудиту.

Наприклад, WMS може забезпечити прозорість залишків запасів організації в будь-який час і в будь-якому місці, як на складі, так і в дорозі. Вона також може керувати операціями з ланцюгами постачання від виробника або гуртового постачальника до складу, а потім до роздрібного споживача або дистриб'ютора. WMS часто використовується разом із системою управління транспортом (TMS)

або системою управління запасами.

Типи систем управління складом

Типи системи управління складом зазвичай залежить від розміру та сфери діяльності організації. Вони можуть бути самостійними системами або модулями у більш масштабній системі управління ресурсами підприємства (ERP) або частиною пакету програм.

Вони також можуть бути різної складності. Деякі невеликі організації можуть використовувати просту програму для друку документів або електронні таблиці, але більшість великих організацій - від малого та середнього бізнесу до корпоративних компаній - використовують складне програмне забезпечення WMS. Деякі налаштування WMS розроблені спеціально для організацій різного розміру, і багато постачальників мають версії продуктів WMS, які можуть масштабуватися в різних по розмірах організаціях. Деякі організації створюють власну WMS з нуля, але частіше купляють WMS у постачальника.

WMS також може бути розроблена або налаштована для конкретних вимог організації; Наприклад, підприємства які займаються продажами в інтернеті можуть використовувати WMS, що має іншу функціональність, ніж підприємства роздрібної торгівлі будівельними матеріалами. Крім того, WMS також може бути розроблена або налаштовано спеціально по типах товарів, які продає організація; наприклад, роздрібний продавець спортивних товарів матиме іншу специфіку, ніж постачальник продуктів харчування.

Переваги системи управління складом

Хоча WMS є складною і дорогою для впровадження та експлуатації, підприємства отримують переваги, які можуть виправдати складність та витрати. Впровадження WMS може допомогти організації скоротити витрати на оплату праці, покращити точність складських запасів, підвищити гнучкість та оперативність, зменшити помилки у відборі та відвантаженні товарів та покращити обслуговування клієнтів. Сучасні системи управління складом працюють із даними в режимі реального часу, що дозволяє організації керувати актуальною інформацією про такі процеси, як замовлення, відвантаження, надходження та

будь-які рухи товарів.

Особливості систем управління складом

Багато функцій є загальними для продуктів WMS, включаючи такі:

- Структура складу, яка дозволяє організаціям налаштовувати робочий процес та логіку відбору, так щоб бути певними, що товари оптимально розподіляються на складських площах. WMS використовує механізм, який максимізує місце для зберігання та показує відхилення сезонних запасів.
- Відстеження наявності запасів, застосовуючи передові системи відстеження, включаючи радіочастотну ідентифікацію(RFID), автоматичну ідентифікацію та збір даних(AIDC) та сканери штрих-кодів, що дає можливість легко знайти товари, коли їх потрібно перемістити.
- Вибір та пакування товарів, включає вибір партії та серій. Складські працівники також можуть використовувати найбільш багатофункціональне зонування та функції перехресних завдань, щоб найбільш ефективно керувати завданнями збірки та пакування.
- Управління працею, допомагає менеджерам складу контролювати ефективність роботи працівників за допомогою ключових показників ефективності (KPI), які вказують на працівників, які працюють краще або гірше стандартів.
- Звітність, яка допомагає менеджерам проаналізувати продуктивність складських операцій та знаходити сфери для покращення.

WMS та IOT

Приєднані пристрої та датчики у продуктах та матеріалах допомагають організаціям забезпечити можливість виробляти та передавати потрібну кількість товарів за правильною ціною в потрібне місце в потрібний час. Всі ці функції потрапляють в Інтернет речі(IoT).

Такі дані IoT можуть інтегруватися в WMS, щоб допомогти керувати маршрутизацією продуктів з точки збору до кінцевої точки. Така інтеграція дає

організаціям змогу розвивати послідовні ланцюжки постачань, а не групові. Послідовні ланцюги постачання залежить від споживчого попиту, що дозволяє організувати більшу гнучкість та оперативність, тоді як групове постачання формується довгостроковими прогнозами попиту споживачів.

Найпопулярніші постачальники WMS

Різні постачальники програмного забезпечення продають системи управління складом. IBM, Microsoft, Oracle та SAP мають продукти або модулі WMS у комплексних пакетів ERP. Але найбільш доступними на ринку України є програмні продукти BAS до складу яких входить модуль WMS це BAS Управління торгівлею та BAS ERP. [9]

## Висновки до розділу 1

У першому розділі було проведено детальний аналіз сучасного стану збору та обробки даних, з акцентом на методи, використовувані в Інтернеті та автоматизовані системи, що знаходять застосування у сфері електронної комерції. Було розглянуто загальні поняття, що стосуються збору даних, включаючи веб-парсинг, використання API, ручний збір даних, інтернет-опитування та порівняння цих методів. Кожен метод має свої переваги та недоліки, і їх вибір залежить від конкретних цілей та умов збору інформації.

Також у цьому розділі було оглянуто сучасні системи автоматизованої обробки даних, включно з системами планування ресурсів підприємства (ERP), системами управління взаємовідносинами з клієнтами (CRM) та системами управління складом. Кожна з цих систем грає важливу роль у ефективному управлінні даними та ресурсами в контексті електронної комерції. Розгляд цих систем дозволяє зрозуміти, як технологічні інновації сприяють оптимізації процесів збору та аналізу даних, підвищуючи ефективність та конкурентоспроможність підприємств.

# РОЗДІЛ 2. РОЗРОБКА АВТОМАТИЗОВАНОЇ СИСТЕМИ ЗБОРУ ТА ОБРОБКИ ДАНИХ

## 2.1. Принципи розробки програмного забезпечення

Коли ми думаємо про принципи проектування програмного забезпечення для технологій, ми часто думаємо про використання кольору на веб-сайті або макет екрана нашого ERP програмне забезпечення. Однак дизайн йде глибше, оскільки він формує інтерфейс із основною часто складною технологією. Важливо зробити цю технологію доступною для користувачів простим і зрозумілим способом. Крім того, хороший дизайн програмного забезпечення може зробити складну технологію простою на перший погляд. Якщо нам це вдасться, більшість проектних завдань виконано успішно.

Технологічні цілі для принципів розробки програмного забезпечення

1. Ставте людей на перше місце

Проектуйте системи відповідно до потреб клієнтів, відвідувачів і користувачів, розумійте та поважайте їх. Ось чому нам потрібно приділити час, щоб дізнатися про людину в цілому — її емоційні, фізичні та технічні потреби. Ми розробляємо наше програмне забезпечення для кінцевого результату.

Зміцнення людей – технологічний зв’язок

Розробка програмного забезпечення повинна отримувати найкраще від технологій і найкраще від людей. Дизайн програмного забезпечення має бути створений таким чином, щоб людина і машина підсилювали одна одну.

2. Уникнення складності

У процесі проектування ми повинні зосередитися на використанні якомога меншої кількості технологій для вирішення проблеми. Адже чим більше технологій, тим більше людям доводиться думати. Комплексні рішення призводять до більшої кількості несправностей і більшої ймовірності людської помилки.

3. Розробка програмного забезпечення в контексті: пов’язуйте системи, де це можливо

Ми не тільки розробляємо дизайн для нашої власної системи, але також маємо враховувати весь досвід користувача, а також залучену інфраструктуру та

процеси. Ми повинні завжди використовувати існуючі системи та рішення. Виберіть для автоматичне посилання, навіть якщо вони належать іншій організації. Посилання, створені вручну, призводять до помилок друку та часто виявляються невдалими.

4. Принципи проектування помилок

Дизайн програмного забезпечення повинен зробити технологію можливою. Автомобіль може виглядати дуже привабливо, але якщо ми не можемо його завести, це марно. Навіть якщо технологія виходить з ладу, дизайн має бути створений таким чином, щоб стали доступними альтернативи. Ми повинні з повагою ставитися до помилок і точно вказувати, що це за помилка та як її виправити. Дизайн програмного забезпечення повинен пропонувати користувачам рішення.

Принципи розробки програмного забезпечення для інтерфейсів

1. Простота

Нам потрібно не лише забезпечити цінні результати на наших екранах, але й бути чіткими та недвозначними. Переконайтеся, що можливості системи додають цінності, а не ускладнюють. Пам’ятайте, що хороший дизайн програмного забезпечення – це якомога менше дизайну.

2. Гнучкість

Дайте декілька варіантів. Подумайте про те, хто, чому, як, де і коли хтось використовує нашу систему або веб-сайт. Тому треба дбати про різне досвід користувача щоб користувачі мали вибір. Також пам’ятайте про використання смартфонів на додаток до ноутбуків.

3. Інформація

Технологія повинна інформувати та створювати спокій. Нічого не припускайте. Переконайтеся, що мета системи зрозуміла, що зміст, структура та послідовність мають сенс і не передають шум на всі органи чуття. Забезпечте дизайн, який дозволить людям знати, де вони знаходяться в системі. Будьте своєчасними, передбачуваними та точними.

4. Профілактика

Надайте прості інструкції та обережно керуйте користувачами під час взаємодії з системою. Допоможіть їм уникнути помилок при введенні даних за допомогою продуманого дизайну форми.

5. Толерантність

Якщо система або веб-сайт призначені для великої (невідомої) групи людей, ми повинні бути гостинними та не дискримінувати. Ось чому ми повинні звернути увагу на люди з різним фізичним, психічним здоров'ям, культурним або соціальним походженням.

6. Зручність

Ми повинні знати, що ми не накладаємо жодних вимог чи обмежень на наших користувачів. Людям не потрібно шукати чи довго думати, щоб щось зробити в нашій системі. Система має бути ефективною та ефективною у використанні.

7. Послідовність

Ми можемо отримати значну користь від дотримання стандартів, загальних вказівок, передового досвіду і конвенцій. Це створює знайоме відчуття, тому користувачі швидше повертаються.

8. Прозорість

Ми повинні бути прозорими щодо нашої роботи та ділитися своїми дизайнерськими рішеннями. Зрештою це призводить до кращих конструкцій і систем. [10]

### 2.2 Zakaz.UA і ПрАТ «Оболонь»: Огляд та аналіз діяльності

Для розуміння специфіки розробки програмного забезпечення потрібно зосередитися на аналізі двох ключових гравців українського ринку – онлайн-супермаркету Zakaz.UA та одного з найбільших виробників пива та безалкогольних напоїв в Україні – ПрАТ «Оболонь». Ці компанії відіграють значну роль у своїх сегментах ринку і є яскравими прикладами впровадження інноваційних підходів у бізнес-процесах та маркетингових стратегіях.

Zakaz.UA виступає не лише як платформа для онлайн-покупок, а й як зразок ефективного використання цифрових технологій для оптимізації клієнтського

досвіду. З іншого боку, ПрАТ «Оболонь» є визнаним лідером у виробництві пива та напоїв, демонструючи високі стандарти якості та інновації у виробничих процесах.

Дослідження діяльності цих компаній надає цінне уявлення про тенденції та виклики, що стоять перед сучасними підприємствами в Україні, а також про ролі, які цифровізація та інновації відіграють у їхньому розвитку.

### 2.2.1 Zakaz.UA

Zakaz.ua — український сервіс доставки продуктів, що входить до групи компаній Zakaz Global. Замовлення здійснюється за допомогою застосунку Zakaz.ua на iOS та Android, або через сайт.

Сервіс працює на ринку України з 2010 року та надає послуги замовлення та доставки продуктів харчування з продуктових мереж роздрібної торгівлі та ринків: Metro, Ашан, Novus, МегаМаркет, ЕКО Маркет, Таврія В, Varus, Восторг, супермаркет Харків, WineTime та ринок Столичний у Києві.

Станом на травень 2023 року компанія працювала в 13 містах України (Київ, Харків, Дніпро, Одеса, Львів, Запоріжжя, Вінниця, Житомир, Полтава, Кривий Ріг, Чернівці, Івано-Франківськ, Рівне), а також у Молдові (Кишинів, Бєльці).

У січні 2022 року було оголошено, що іспанська компанія Glovo домовилась про поглинання сервісу. Та акціонер компанії Степан Черновецький заявив, що угоду не було закрито. [11]

### 2.2.2 ПрАТ «Оболонь»

Структура корпорації «ОБОЛОНЬ» формувалася довгі роки під впливом стратегії розвитку, що направлена на диверсифікацію виробництва, перехід на сировину власного виробництва, інноваційний підхід, абсолютну екологічну безпеку та повну соціальну відповідальність.

Корпорація «ОБОЛОНЬ» об'єднує 10 підприємств по всій Україні. Загалом у корпорації працює майже 4 тисячі людей.

У 1998 році «ОБОЛОНЬ», першою у харчовій галузі України, отримала сертифікат на систему управління якістю ISO 9001. Через 10 років корпорація першою сертифікувала одразу чотири системи управління.

На даний час в корпорації сертифіковані і діють:

Система управління якістю (ISO 9001:2015) — система взаємопов'язаних, орієнтованих на задоволення споживачів процесів, які постійно поліпшуються завдяки лідерству керівництва і залучення персоналу, діючих на підставі фактів, а також взаємовигідних стосунків з постачальниками.

Система управління безпечністю харчових продуктів (ISO 22 000:2018) — попереджувальна система для забезпечення безпечності харчових продуктів (постійний аналіз небезпечних чинників та перевірка критичних контрольних точок на всіх етапах виробництва).

Система екологічного керування (ISO 14 001:2015) — розробка та запровадження екологічної політики компанії, керування її екологічними аспектами.

Система управління безпекою та гігієною праці (ISO 45 001:2018) — дає можливість організації управляти ризиками в області безпеки і гігієни праці і покращувати свої показники в цій області.[12]

### 2.3 Огляд технічного завдання від ПрАТ «Оболонь»

У рамках проходження переддипломної та виробничої практики, я отримав унікальну можливість працювати над проектом, запропонованим ПрАТ «Оболонь». ПрАТ «Оболонь», що є одним із провідних виробників напоїв в Україні, виявило потребу в розробці технічного рішення, яке б відповідало сучасним вимогам ефективності та інноваційності.

Технічне завдання полягає у створенні інструменту для парсингу сайту zakaz.ua з наступними вимогами:

1. Ціль: Збір інформації про ціни на товари в магазинах, з можливістю вибору товарів для парсингу:
    - Парсити всі товари у вказаних категоріях;
    - Можливість вибрати для парсингу тільки певні товари.
2. Обмеження: Парсити інформацію лише про товари, які є в наявності.
3. Формат результатів: Вихідні дані парсингу повинні бути у форматі xlsx (Excel). Планується налаштування запису даних у базу даних у

майбутньому.

4. Інтересуючі категорії товарів:
    - Алкогольні напої (пиво, сидр, слабоалкогольні напої);
    - Напої (мінеральна вода, солодкі води та напої, енергетики).
5. Необхідні дані для кожного товару:
    - Мережа, де продається товар;
    - Код товару (з посилання на товар);
    - Повна назва товару;
    - Об’єм тари (у мл);
    - Ціна;
    - Акційна ціна (якщо є);
    - Строк дії акції (якщо є).
6. Формат виводу інформації: Дані потрібно записувати накопичувально, зберігаючи історію цін. Кожен запис має містити дату отримання інформації.
7. Контакт для питань: cio@obolon.ua

### 2.4 Розширення технічного завдання та остаточні вимоги до проекту

Під час детального аналізу проекту та врахування потенціалу моєї кваліфікаційної роботи, я виявив можливість доповнення проекту новими функціями. Ці доповнення включають:

- Розширення зібраних даних про товар (категорія товару, бренд, url товару)
- Додання методів аналізу і візуалізації даних (Співвідношення ціна/обсяг, середня ціна за брендами, порівняльний аналіз цін за брендами, середня ціна для продуктів з та без промоцій, середня ціна за брендами та категоріями, залежність цін за одиницю об’єму)

Після розширення технічного завдання остаточні вимоги до програмного забезпечення виглядають так:

Технічні вимоги

*Таблиця 2.1. Технічні вимоги проекту*

| Назва вимоги | Перелік завдань |
|---|---|
| Апаратна інфраструктура | Мінімальні вимоги до обсягу оперативної пам'яті та обчислювальної потужності для ефективного виконання завдань. |
| Програмна інфраструктура | - Використання мови програмування Python для розробки скриптів та забезпечення зручності управління системою.<br>- Застосування PyQt та Qt Designer для створення графічного інтерфейсу з елементами керування.<br>- Використання бібліотек matplotlib та Pandas для візуалізації даних та забезпечення зрозумілого аналізу. |
| Технології | - Застосування бібліотек BeautifulSoup для реалізації скриптів парсингу даних з веб-сайту zakaz.ua.<br>- Використання бібліотек pandas та openpyxl для ефективного зберігання та організації даних у файлі Excel. |

Функціональні вимоги

*Таблиця 2.2. Функціональні вимоги проекту*

| Назва вимоги | Перелік завдань |
|---|---|
| Збір та обробка даних | - Можливість автоматичного збору даних з сайту zakaz.ua, включаючи інформацію про ціни, бренди та інші параметри.<br>-Автоматизована обробка отриманих даних для створення зручної та легкозрозумілої структури. |
| Графічний інтерфейс: | - Ефективний графічний інтерфейс, який містить елементи керування для запуску та контролю парсингу, а також відображення результатів. |
| Візуалізація даних | - Можливість побудови графіків та діаграм для кращого розуміння цінової динаміки та популярності брендів. |

## 2.5 Проектування автоматизованої системи збору та обробки даних

**Архітектура системи:**

Розробка архітектури системи включає в себе розподіл функцій між різними компонентами. Це може бути розділено на:

- Модуль інтерфейсу, взаємодія з яким забезпечить користувачам зручний доступ до функціоналу системи.

Обґрунтування вибору: Створення модулю інтерфейсу є ключовим для забезпечення зручного та ефективного взаємодії користувачів з системою. Вибір використання графічного інтерфейсу дозволяє створити інтуїтивно зрозумілий інтерфейс, який спростить введення команд, отримання результатів та взаємодію з іншими модулями системи.

- Модуль веб-парсингу, що відповідає за отримання та обробку інформації з сайту zakaz.ua.

Обґрунтування вибору: Модуль парсингу визначається як необхідний для отримання та обробки інформації з сайту zakaz.ua. Використання парсингу дозволяє ефективно та автоматизовано отримувати дані з веб-сайту, надаючи системі необхідну інформацію для подальшого використання.

- Модуль візуалізації для ефективного представлення даних.

Обґрунтування вибору: Модуль візуалізації є важливим для створення зручного та ефективного представлення даних. Використання візуалізаційних елементів допомагає користувачам швидше розуміти та аналізувати інформацію, що полегшує прийняття рішень.

- Механізм зберігання, який відповідає за зберігання даних у файлі Excel.

Обґрунтування вибору: Механізм зберігання, який використовує файл Excel, обрано з урахуванням зручності та поширеності цього формату для зберігання даних. Файл Excel легко читається та редагується багатьма програмами, що робить його практичним для використання в роботі з отриманими даними.

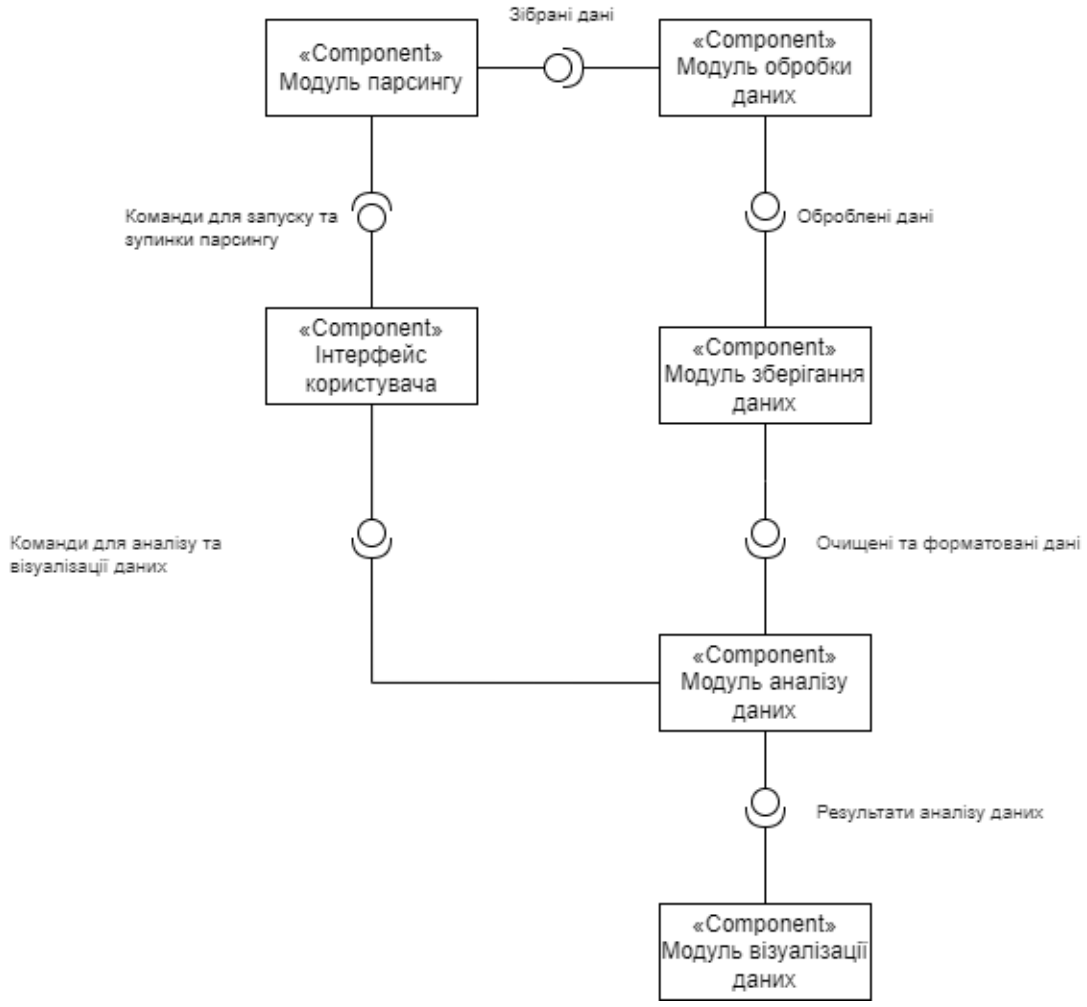



Рис.2.1 – Діаграма компонентів проекту

**Графічний інтерфейс:**

Створення графічного інтерфейсу використовує технології PyQt та Qt Designer для забезпечення зручної взаємодії користувачів. Використання PyQt дозволяє легко інтегрувати графічний інтерфейс зі скриптами на Python, а Qt Designer надає можливість швидко створювати та налаштовувати елементи керування, такі як кнопки для запуску парсингу та відображення результатів. Це забезпечить зручність в користуванні та ефективну роботу з системою.

**Візуалізація даних:**

Візуалізація даних використовує бібліотеки, такі як matplotlib та Plotly, для створення графіків та діаграм для кращого розуміння та аналізу даних. Використання цих бібліотек дозволяє легко побудувати різноманітні візуальні елементи, такі як графіки цін чи популярності брендів, щоб користувачі могли

швидко отримати уявлення про дані.

**Скрипти для Парсингу:**

Розробка скриптів для парсингу базується на використанні Python як мови програмування, а також на бібліотеках BeautifulSoup для роботи з HTML. Python володіє зручними засобами для реалізації скриптів парсингу, а бібліотеки забезпечують ефективний та структурований доступ до інформації на веб-сайті zakaz.ua.

**Зберігання даних у файлі Excel:**

Механізм зберігання даних у файлі Excel використовує бібліотеки pandas та openpyxl для ефективної організації та зберігання даних. Pandas дозволяє легко створювати та редагувати DataFrame, а openpyxl забезпечує зручний доступ до Excel-файлів. Використання цих бібліотек спрощує роботу з отриманими даними та забезпечує зручний аналіз результатів.

## 2.6 Розробка програмного забезпечення

Щоб створити програмний продукт згідно технічного завдання та діаграми компонентів було розроблено модулі системи, які можна переглянути у додатку Б:

**Модуль парсингу**

Функції модуля:

- Завантаження сторінок:
  - load_page(product_category, shop, page): Завантажує HTML-контент сторінки з товарами за заданою категорією та магазином.
- Парсинг сторінок:
  - parse_page(text, shop): Розбирає HTML-контент сторінки та виділяє основні блоки з товарами.
  - parse_block(block, shop): Парсить інформацію про товар з окремого блоку: код товару, URL, назву, ціну, знижку, термін дії знижки, бренд, об'єм.
- Обробка деталей товару:
  - get_page_content(url): Завантажує HTML-контент сторінки товару.

- parse_product_details(html_content): Витягує детальну інформацію з сторінки товару: назву, ціну, знижку, термін дії знижки, бренд, об'єм.

**Модуль обробки даних**

Функції модуля:

- Збереження результатів:
  - save_result(category, shop): Зберігає відпарсені дані в Excel-файл zakaz.xlsx

**Модуль зберігання даних**

Бібліотека: openpyxl

Функціонал:

- Створення Excel-файлу.
- Додавання заголовків.
- Запис даних у рядки.
- Збереження файлу.

**Модуль інтерфейсу користувача**

Реалізує графічну оболонку програми.

Надає можливість користувачу:

- Вибирати тип парсингу (повний або вибірковий).
- Вибирати магазини та категорії товарів для парсингу.
- Запускати та зупиняти процеси парсингу.
- Переглядати хід виконання парсингу за допомогою прогрес-барів.
- Налаштовувати автоматичний запуск парсингу за розкладом.
- Вибирати методи аналізу даних.
- Переглядати результати аналізу у вигляді графіків та діаграм.

**Модуль аналізу даних**

Здійснює обробку та аналіз зібраних даних.

Містить методи аналізу даних такі як:

- Співвідношення ціна/обсяг.
- Середня ціна за брендами
- Порівняльний аналіз цін за брендами
- Середня ціна для продуктів з та без промоцій
- Середні ціни за брендами та категоріями
- Залежність ціни за одиницю об'єму від об'єму

**Модуль візуалізації даних**

Відповідає за створення візуального представлення результатів аналізу.

Matplotlib генерує графіки та діаграми, відповідно до методів аналізу даних, що дає змогу візуалізувати результати.

**Збірка програми**

Програма зібрана за допомогою pyinstaller. Це означає, що вона є самодостатнім виконавчим файлом, який не потребує Python або будь-яких інших бібліотек для запуску.

Програма містить такі файли:

- favicon.ico: Файл іконки програми.
- main.py: Головний файл Python, який запускає програму.
- ParserWorkers.py: Файл, який містить код для фонових процесів парсингу.
- parsermodule.py: Файл, який містить код для парсингу даних з веб-сайтів.
- resources.qrc: Файл ресурсів, який містить зображення та інші дані.
- resources_rc.py: Файл, який генерується з resources.qrc.
- ZakazuaParser.py: Файл, який містить код для парсингу даних з веб-сайту Zakaz.ua.

- ZakazuaParser.ui: Файл інтерфейсу користувача для парсингу Zakaz.ua.
- Бібліотеки Python: Програма також містить всі бібліотеки Python, які необхідні для її роботи.

### 2.7 Інструкція користувача

Після завантаження файлу програму запустіть файл, щоб почати роботу з програмою. Після запуску програми ви побачите головне вікно інтерфейсу користувача.

Інтерфейс користувача програми поділений на кілька основних секцій:

1. Меню - містить пункти меню для доступу до різних функцій програми. А саме: парсинг по всім мережам, вибірковий парсинг, аналіз та налаштування
2. Робоча область - відображає поточний стан програми та дані, з якими ви працюєте.

Пункт меню «Парсинг всіх мереж» містить інтерфейс для вибору параметрів парсингу даних з усіх доступних мереж.

Елементи інтерфейсу:

- Вибір категорії товару – елемент, який потрібен для обирання, яку категорію товару будуть парсити по всім мережам.
- Кнопка «Парсинг» - запускає парсер з обраними категоріями товару.

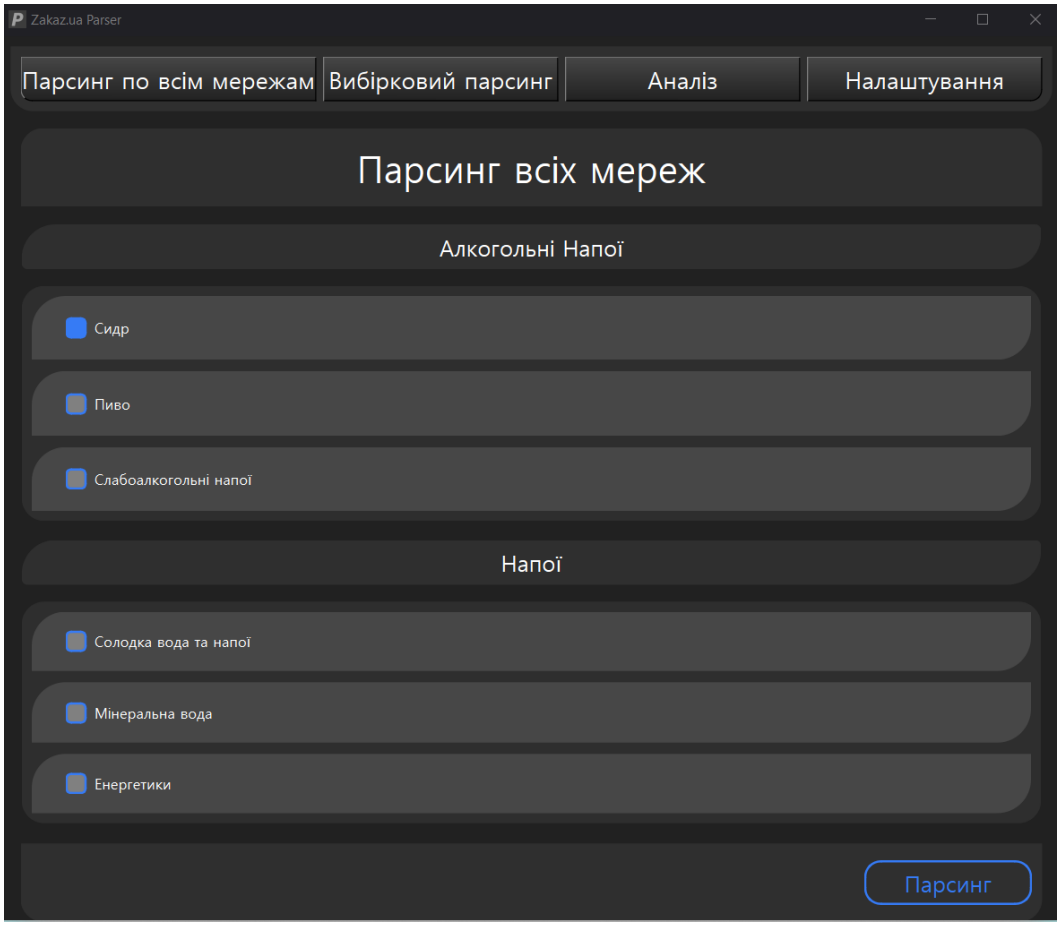



*Рис.2.2 – Вигляд вікна програми «Парсинг всіх мереж»*

Пункт меню «Вибірковий парсинг» містить інтерфейс для вибору параметрів парсингу даних з певних мереж.

Елементи інтерфейсу:

- Вибір мережі – елемент, який потрібен для обрання мережі парсингу.
- Вибір категорії товару – елемент, який потрібен для обирання, яку категорію товару будуть парсити по всім мережам.
- Кнопка «Парсинг» - запускає парсер з обраними категоріями товару.

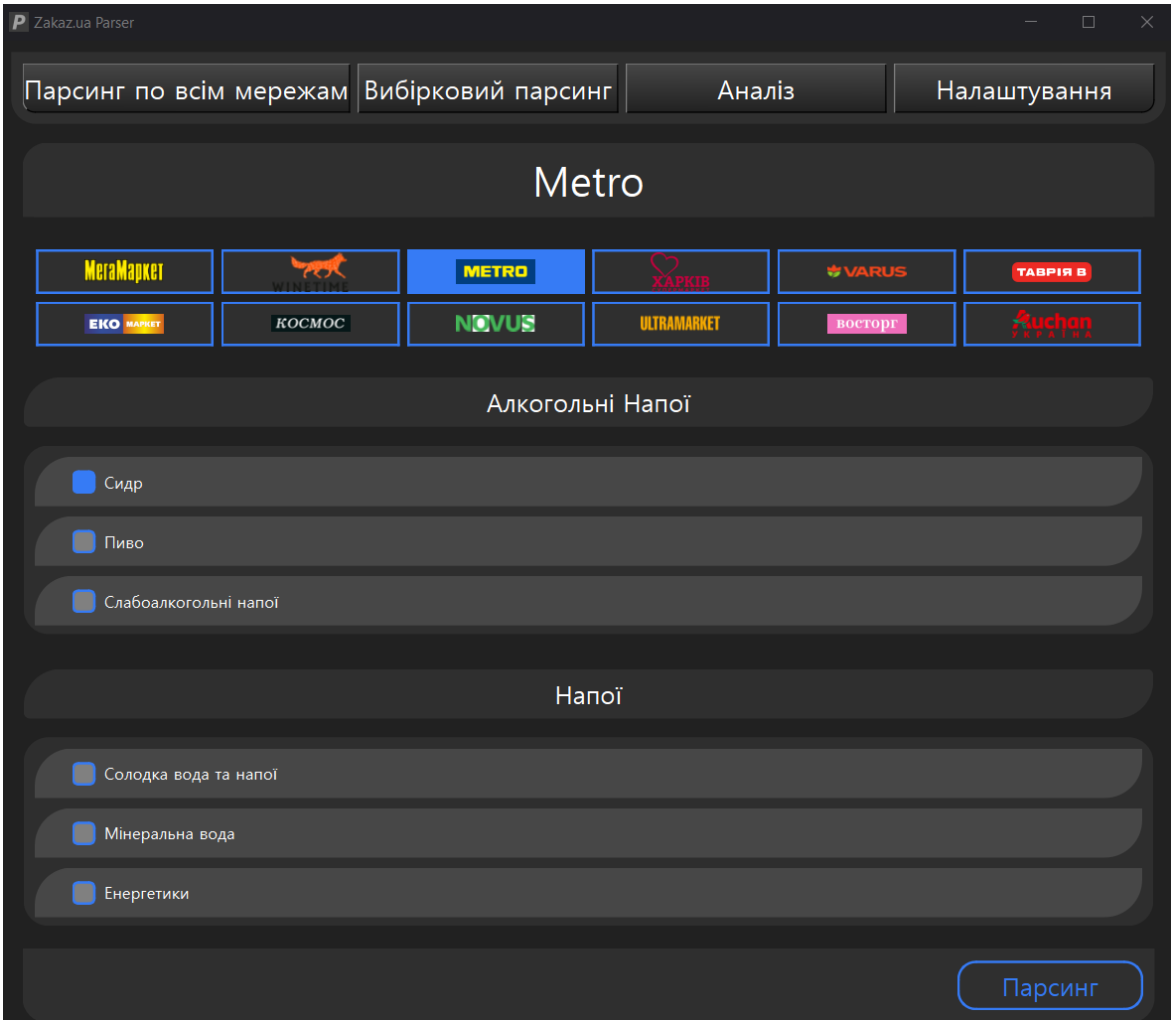



*Рис.2.3 – Вигляд вікна програми «Вибірковий парсинг»*

Пункт меню «Аналіз» містить інтерфейс для вибору методу аналізу даних. Для того щоб запустити метод аналізу даних потрібно відкрити випадаючий список та обрати потрібний метод. Після цього натиснути на кнопку «Запуск аналізу». Вигляд аналітичних методів є можливість переглянути у додатку Б.

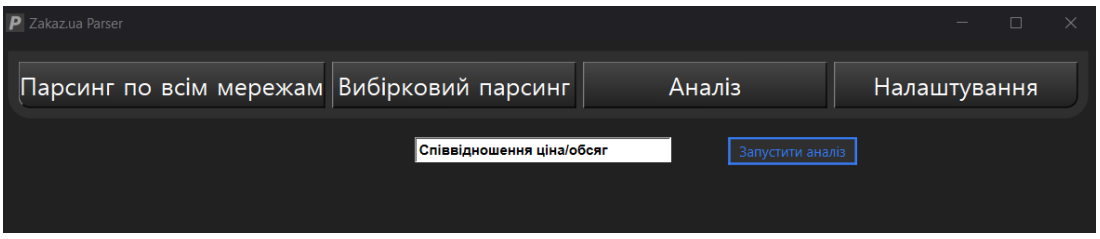



*Рис.2.4 – Вигляд вікна програми «Аналіз»*

Пункт меню «Налаштування» містить інтерфейс для налаштування

автозапуску парсира у певний час.

Для цього потрібно натиснути на чекбокс рядом з цією функцією. Потім обрати певний час запуску та режим парсера.

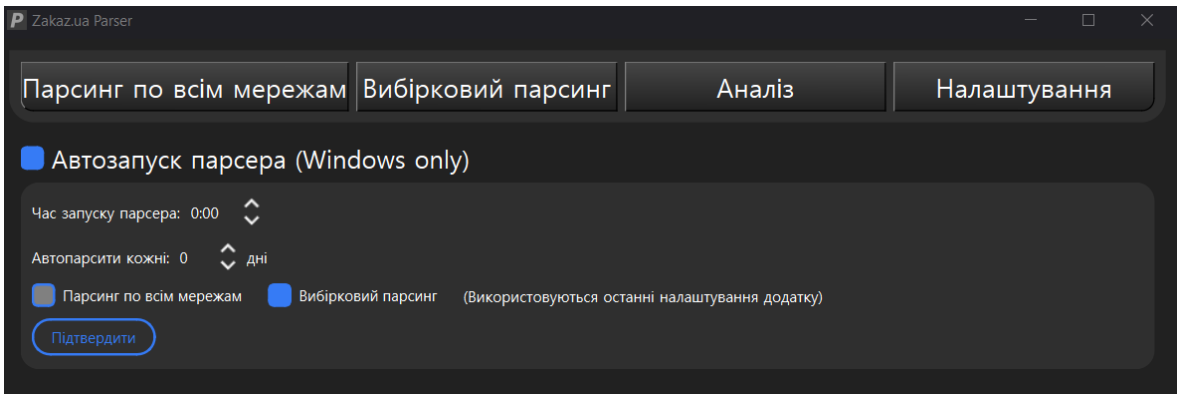



*Рис.2.5 – Вигляд вікна програми «Налаштування»*

Після закінчення парсингу ви побачити в інтерфейсі програми, що прогрес парсингу дійшов до 100%. Дані з програми автоматично додадуться до файлу zakaz.xlsx. Якщо такого файлу нема у наявності то програма автоматично зробить його.

| num | retail | category | sku | vol | price | promo | promo dat | date | brand | url |
|---|---|---|---|---|---|---|---|---|---|---|
| 04820250 | metro | cidr | Сидр Som | 330мл | 31.64 | | | 22.01.202 | Somersby | https://metro.zakaz.ua/uk/products/sidr-somersbi-330ml--04820250942709/ |
| 04820250 | metro | cidr | Сидр Som | 330мл | 31.64 | | | 22.01.202 | Somersby | https://metro.zakaz.ua/uk/products/sidr-somersbi-330ml-ukrayina--04820250942563/ |
| 04820000 | metro | cidr | Сидр Som | 950мл | 66.42 | | | 22.01.202 | Somersby | https://metro.zakaz.ua/uk/products/sidr-somersbi-950ml-ukrayina--04820000458177/ |
| 04820000 | metro | cidr | Сидр Som | 950мл | 67.93 | | | 22.01.202 | Somersby | https://metro.zakaz.ua/uk/products/sidr-somersbi-950ml-ukrayina--04820000455145/ |
| 04820000 | metro | cidr | Сидр Som | 500мл | 43.09 | | | 22.01.202 | Somersby | https://metro.zakaz.ua/uk/products/sidr-somersbi-500ml-ukrayina--04820000453318/ |
| 04820097 | metro | cidr | Сидр Apps | 500мл | 40.93 | | | 22.01.202 | APPS | https://metro.zakaz.ua/uk/products/sidr-apps-500ml-ukrayina--04820097893950/ |
| 04820097 | metro | cidr | Сидр Apps | 500мл | 43.09 | | | 22.01.202 | APPS | https://metro.zakaz.ua/uk/products/sidr-apps-500ml--04820097894568/ |
| 04820000 | metro | cidr | Сидр Som | 500мл | 48.06 | | | 22.01.202 | Somersby | https://metro.zakaz.ua/uk/products/sidr-somersbi-500ml-ukrayina--04820000457217/ |
| 04820250 | metro | low-alcoh | Сидр Som | 330мл | 31.64 | | | 22.01.202 | Somersby | https://metro.zakaz.ua/uk/products/sidr-somersbi-330ml--04820250942709/ |
| 04820250 | metro | low-alcoh | Сидр Som | 330мл | 31.64 | | | 22.01.202 | Somersby | https://metro.zakaz.ua/uk/products/sidr-somersbi-330ml-ukrayina--04820250942563/ |

*Рис.2.6 – Приклад вмісту файлу Excel після парсингу даних*

## 2.8 Порівняльний аналіз проекту з аналогами

Переваги Parser Zakaz.UA:

- Безкоштовний: На відміну від Pricehunter, наш проект є безкоштовним для використання.
- Легкий у використанні: Ми пропонуємо простий та інтуїтивно зрозумілий інтерфейс, який робить парсинг даних доступним для будь-якого користувача.
- Гнучкий: Ви можете кастомізувати процес парсингу даних, вибираючи тип товарів, мережі магазинів та інші параметри.

- Аналіз даних: Наш проект пропонує інструменти для аналізу спарсених даних, що дозволяє вам отримати більш глибоке розуміння ринку.
- Візуалізація даних: Ми пропонуємо інструменти для візуалізації спарсених даних, що робить їх більш зрозумілими та зручними для аналізу.

Недоліки Parser Zakaz.UA:

- Немає можливості порівняння товарів: На відміну від Price.ua, проект не пропонує інструменти для порівняння товарів.

*Таблиця 2.3. Порівняння розробленого проекту з аналогами на ринку*

| **Критерій** | **Parser Zakaz.UA** | **Price.ua** | **Pricehunter** |
|---|---|---|---|
| **Збір інформації** | Наявна | Наявна | Наявна |
| **Легкість використання** | Високий рівень | Високий рівень | Середній рівень |
| **Підтримка форматів даних** | Excel | Excel | Excel |
| **Вартість використання** | Безкоштовно | Безкоштовно | Платно |
| **Кастомізація** | Високий рівень | Середній рівень | Високий рівень |
| **Підтримка ОС** | Windows | Web | Web |
| **Порівняння між товарами** | Ні | Так | За вимогою |
| **Вибір типу товарів для парсингу** | Так | Так | Так |
| **Аналіз даних** | Так | Ні | Ні |
| **Візуалізація даних** | Так | Ні | Ні |

## Висновки до розділу 2

Другий розділ магістерської роботи був присвячений розробці автоматизованої системи збору та обробки даних. Він розпочався з викладення основних принципів розробки програмного забезпечення, які є фундаментальними для створення ефективних та надійних систем. Надалі розділ присвячувався детальному огляду та аналізу діяльності двох компаній - Zakaz.UA та ПрАТ «Оболонь», на основі яких було здійснено розробку конкретного проекту.

Огляд технічного завдання від ПрАТ «Оболонь», а також його розширення та встановлення остаточних вимог до проекту були ключовими для забезпечення того, щоб розроблена система відповідала всім потребам замовника. Процес проектування системи збору та обробки даних був зосереджений на створенні структурованого та ефективного рішення, яке б забезпечило легкість використання та високу продуктивність.

Розробка програмного забезпечення включала в себе створення всіх необхідних компонентів системи, відповідно до заздалегідь встановлених вимог і специфікацій. Також було розроблено детальну інструкцію користувача, що забезпечує зрозумілість та легкість використання системи кінцевими користувачами.

Завершальна частина розділу містила порівняльний аналіз розробленої системи з аналогами, що існують на ринку. Цей аналіз дозволив виявити переваги та потенційні області для подальшого вдосконалення розробленого продукту, відображаючи його конкурентоспроможність та ефективність.

# Висновки

У ході виконання цієї кваліфікаційної роботи було досягнуто поставленої мети: досліджено сучасний стан збору та обробки даних, а також розроблено інформаційну систему автоматизованого збору та обробки даних для ПрАТ «Оболонь» з сайту zakaz.ua. Виконання визначених завдань дозволило отримати наступні висновки:

Методи збору даних в Інтернеті, такі як веб-парсинг, використання API, ручний збір даних та інтернет-опитування, мають свої специфічні переваги та обмеження. Ефективне використання цих методів залежить від конкретних вимог та цілей збору даних.

Сучасні системи автоматизованої обробки даних, включаючи системи планування ресурсів підприємства, системи управління взаємовідносинами з клієнтами та системи управління складом, демонструють високий рівень ефективності та можливість інтеграції в різноманітні бізнес-процеси.

Принципи розробки програмного забезпечення відіграють ключову роль у створенні надійних, ефективних та легких у підтримці інформаційних систем.

Детальний огляд діяльності компаній Zakaz.UA та ПрАТ «Оболонь» дозволив точно визначити вимоги до розроблюваної системи, а також забезпечити її відповідність специфіці цих організацій.

Розроблена програмна система успішно вирішила завдання автоматизованого збору та обробки даних. Проектування та розробка програмного забезпечення були виконані з урахуванням всіх технічних та функціональних вимог.

Інструкція користувача, яка була розроблена для цієї системи, детально описує процес її використання, забезпечуючи легкість та ефективність управління системою.

Порівняльний аналіз розробленого програмного забезпечення з існуючими аналогами на ринку підтвердив його конкурентоспроможність, ефективність та унікальність у контексті специфічних потреб ПрАТ «Оболонь» та можливостей інтеграції з сайтом zakaz.ua.

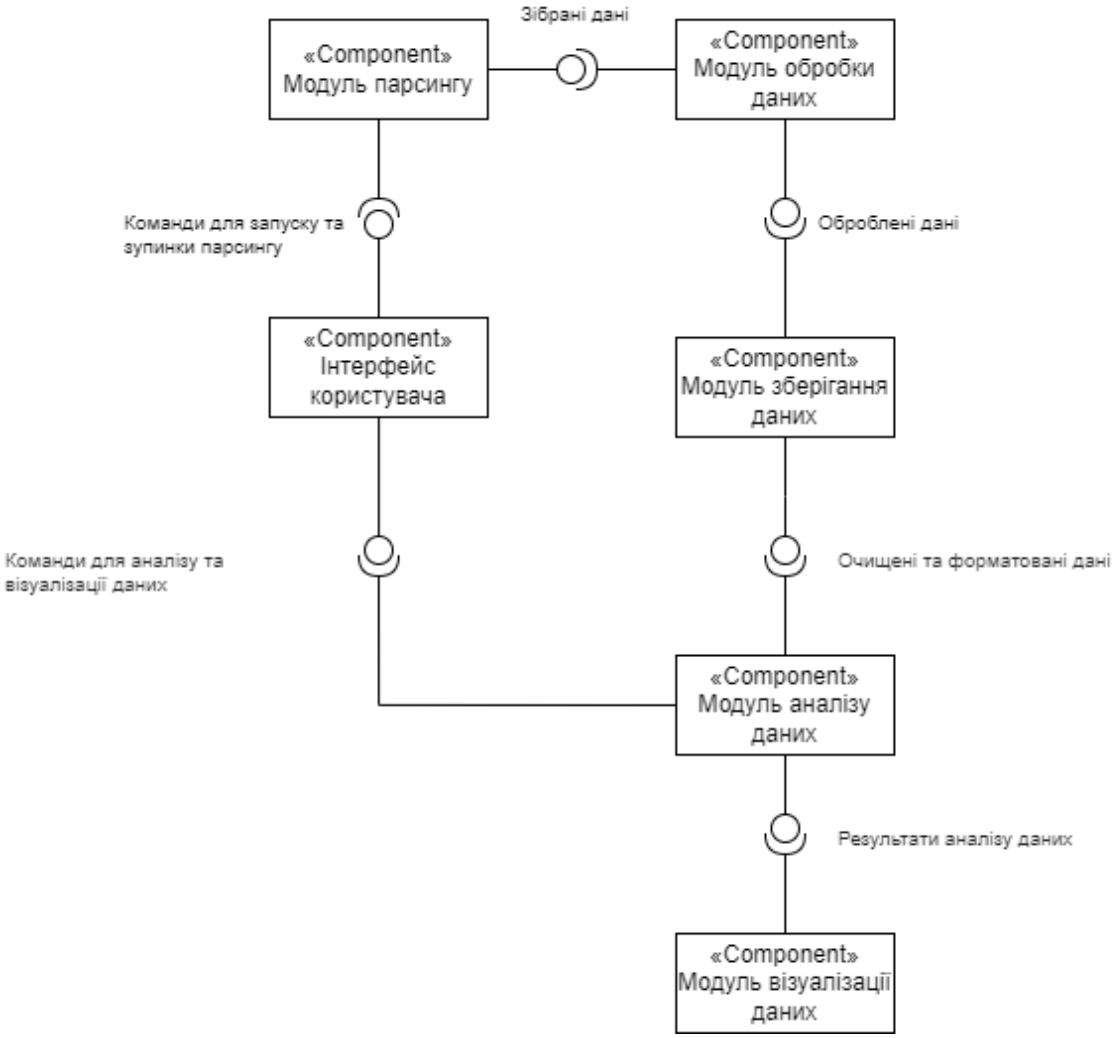
Таким чином, виконана робота демонструє успішне вирішення поставлених завдань, а розроблена система може слугувати надійним інструментом для оптимізації процесів збору та обробки даних у сфері електронної комерції.

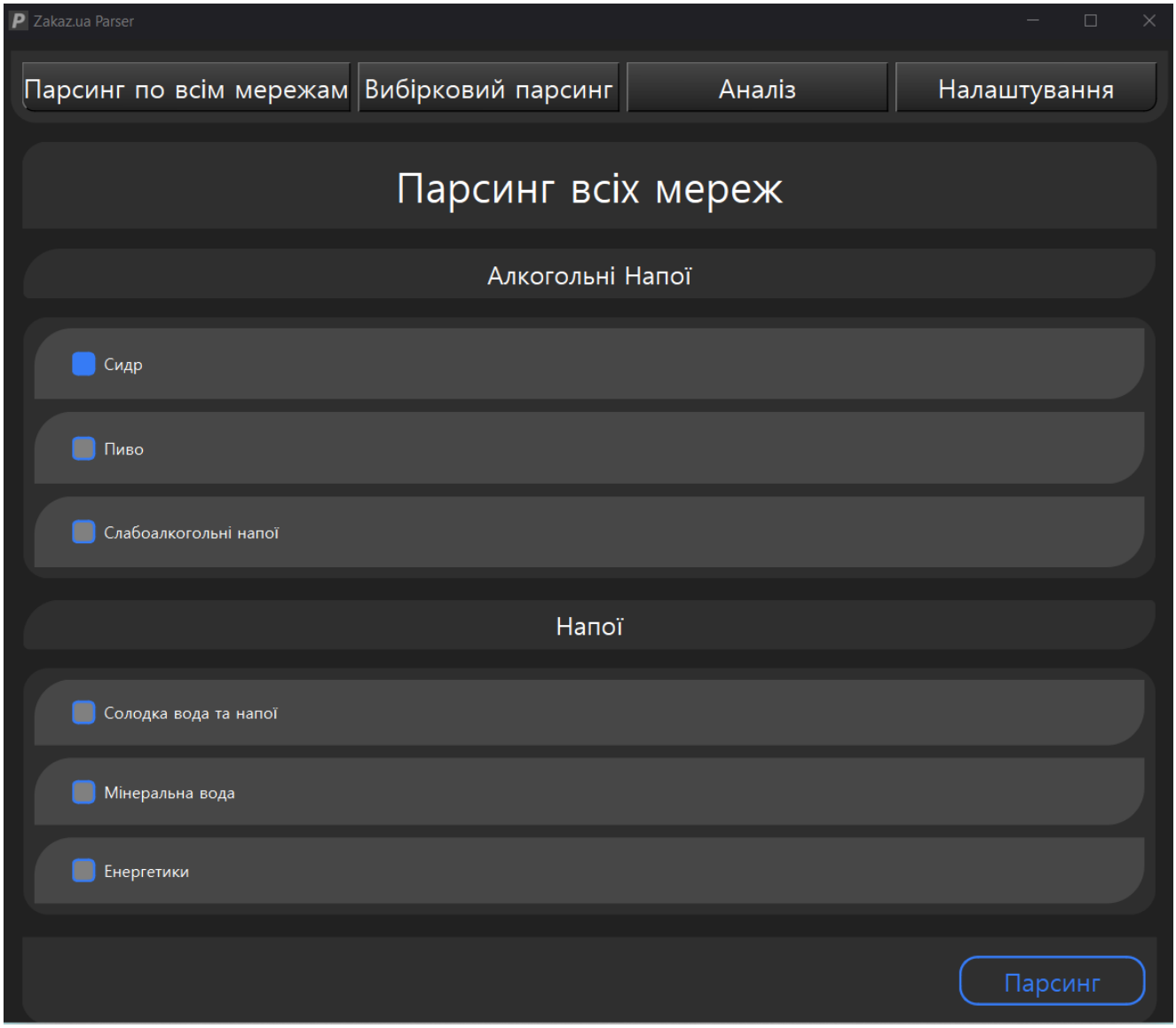
# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

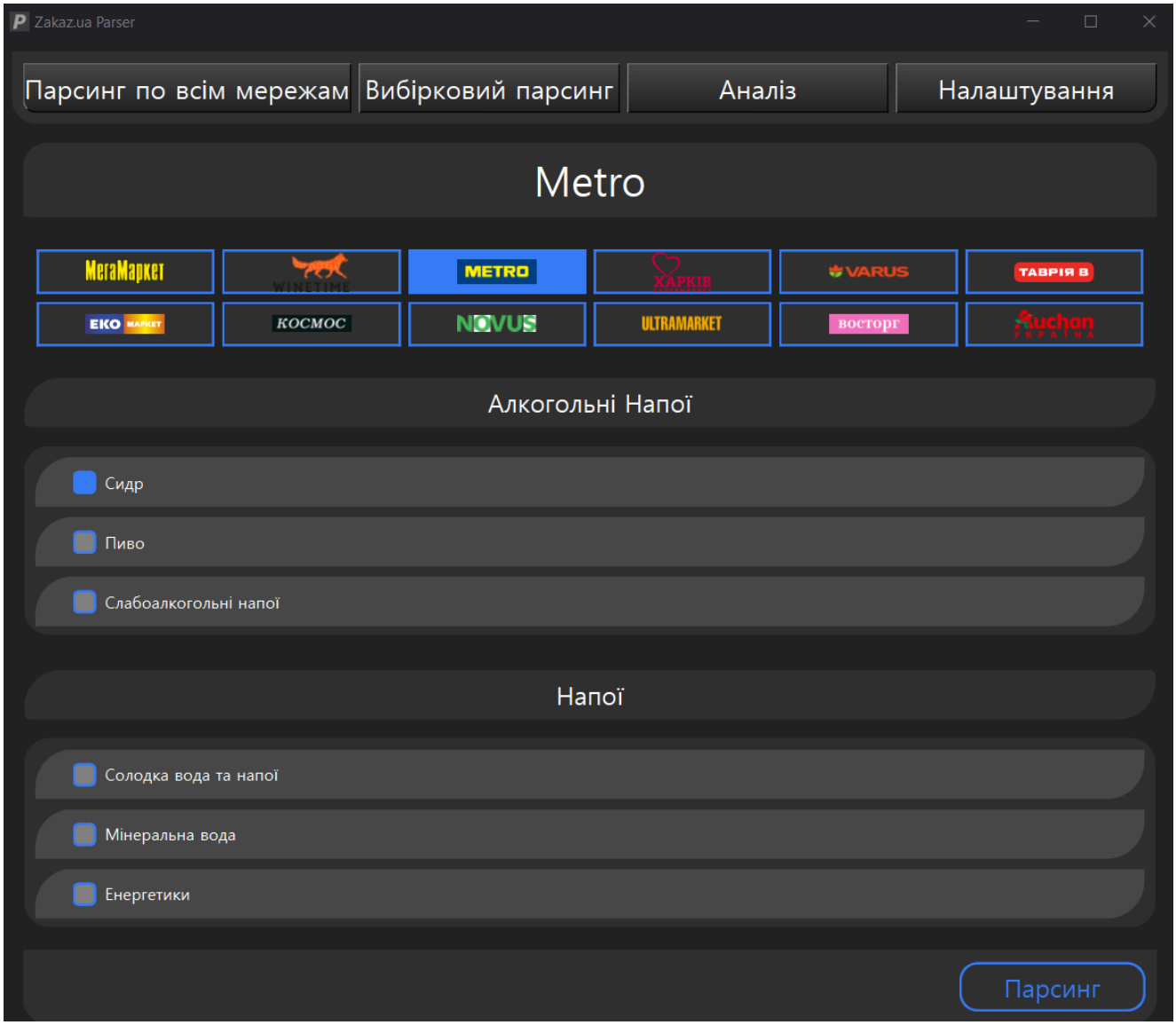

1. Дані (обчислювальна техніка) [Електронний ресурс]. - Режим доступу: https://uk.wikipedia.org/wiki/Дані_(обчислювальна_техніка)
2. Збирання даних [Електронний ресурс]. - Режим доступу: https://uk.wikipedia.org/wiki/Збирання_даних
3. Обробка інформації [Електронний ресурс]. - Режим доступу: https://uk.wikipedia.org/wiki/Обробка_інформації
4. Парсинг сайтів: що це і навіщо він потрібен? [Електронний ресурс]. - Режим доступу: https://web-promo.ua/ua/blog/parsing-sajtov-chto-eto-i-zachem-nuzhen/
5. Що таке API? [Електронний ресурс]. - Режим доступу: https://brainlab.com.ua/uk/blog-uk/shho-take-api
6. MANUAL DATA COLLECTION [Електронний ресурс]. - Режим доступу: https://www.machinemetrics.com/blog/manual-data-collection
7. Що таке ERP? [Електронний ресурс]. - Режим доступу: https://bc.smart-it.com/news-and-articles/what-is-erp/
8. Що таке CRM? [Електронний ресурс]. - Режим доступу: https://sendpulse.ua/support/glossary/crm
9. WMS система управління складом [Електронний ресурс]. - Режим доступу: https://www.netsoft.com.ua/WMS-Systema-upravlinnya-skladom.html
10. Принципи розробки програмного забезпечення та базова технологія [Електронний ресурс]. - Режим доступу: https://uk.itpedia.nl/2022/10/30/de-software-ontwerpprincipes-en-onderliggende-technologie/
11. Zakaz.ua [Електронний ресурс]. - Режим доступу: https://uk.wikipedia.org/wiki/Zakaz.ua
12. ПРаТ «Оболонь» [Електронний ресурс]. - Режим доступу: https://obolon.ua/

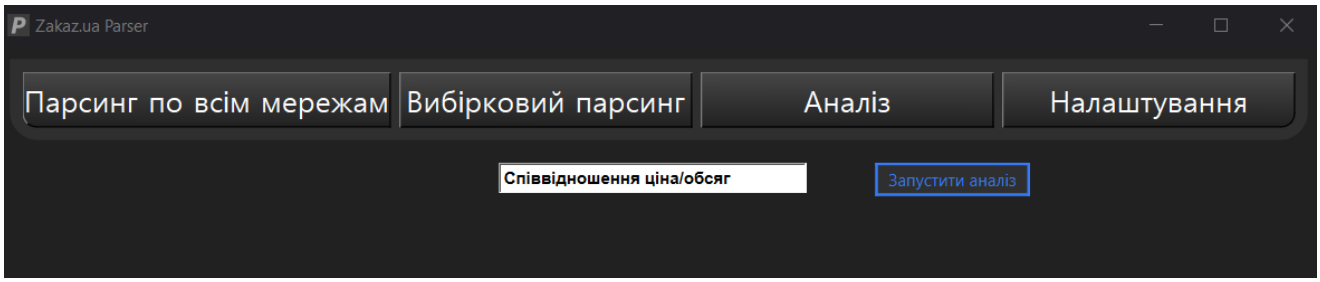
# ДОДАТКИ

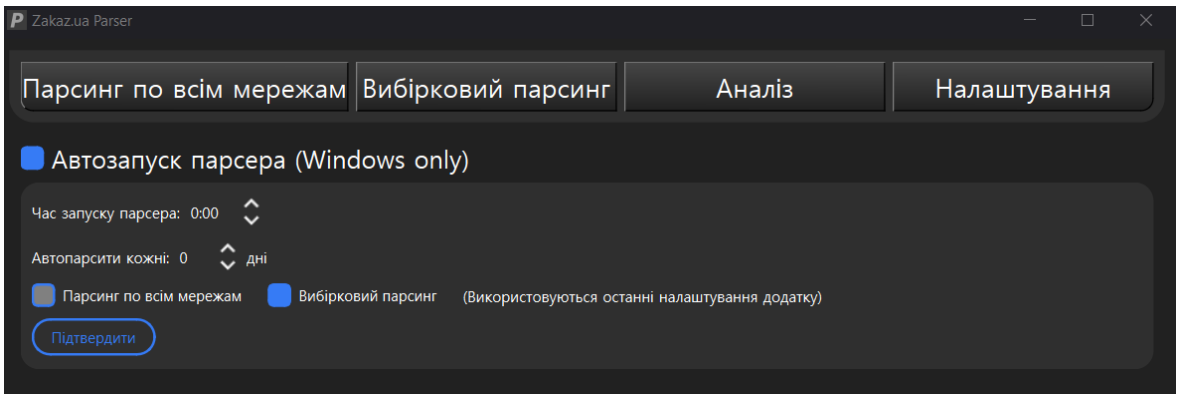
## Додаток А



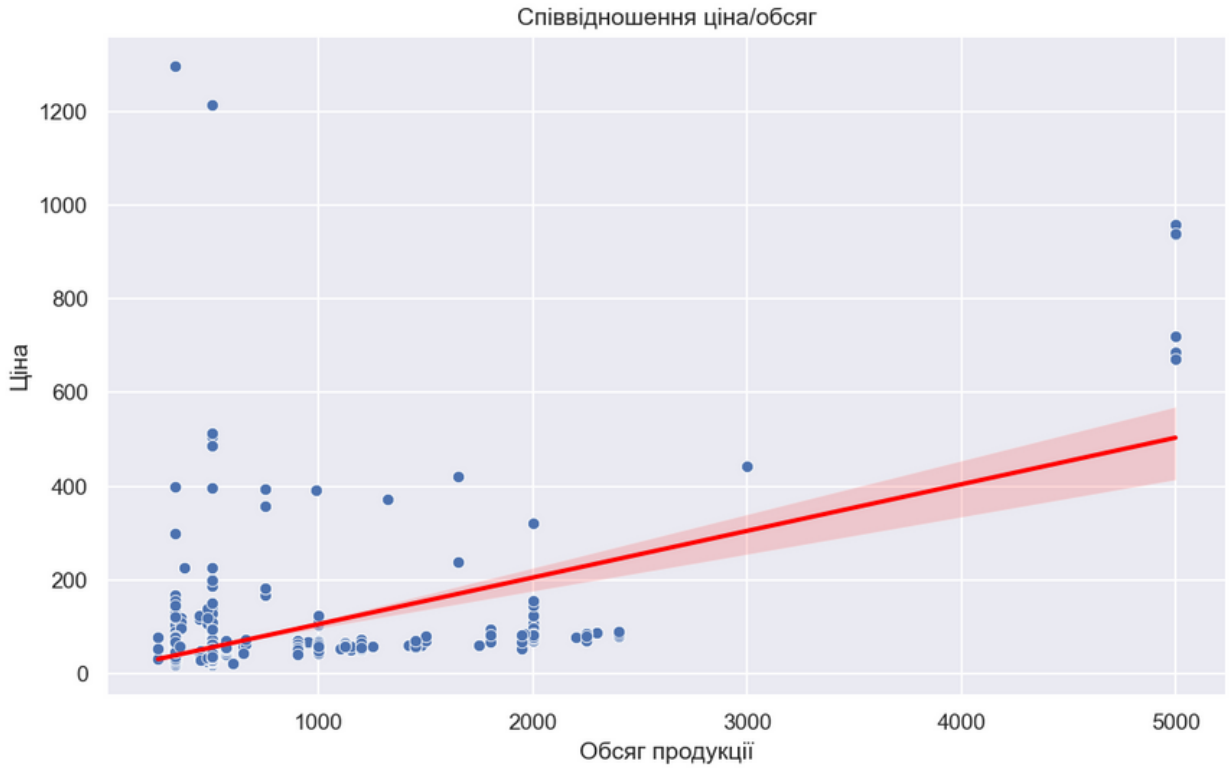

Zakaz.ua Parser
Парсинг по всім мережам
Вибірковий парсинг
Аналіз
Налаштування
Парсинг всіх мереж
Алкогольні Напої
Сидр
Пиво
Слабоалкогольні напої
Напої
Солодка вода та напої
Мінеральна вода
Енергетики
Парсинг

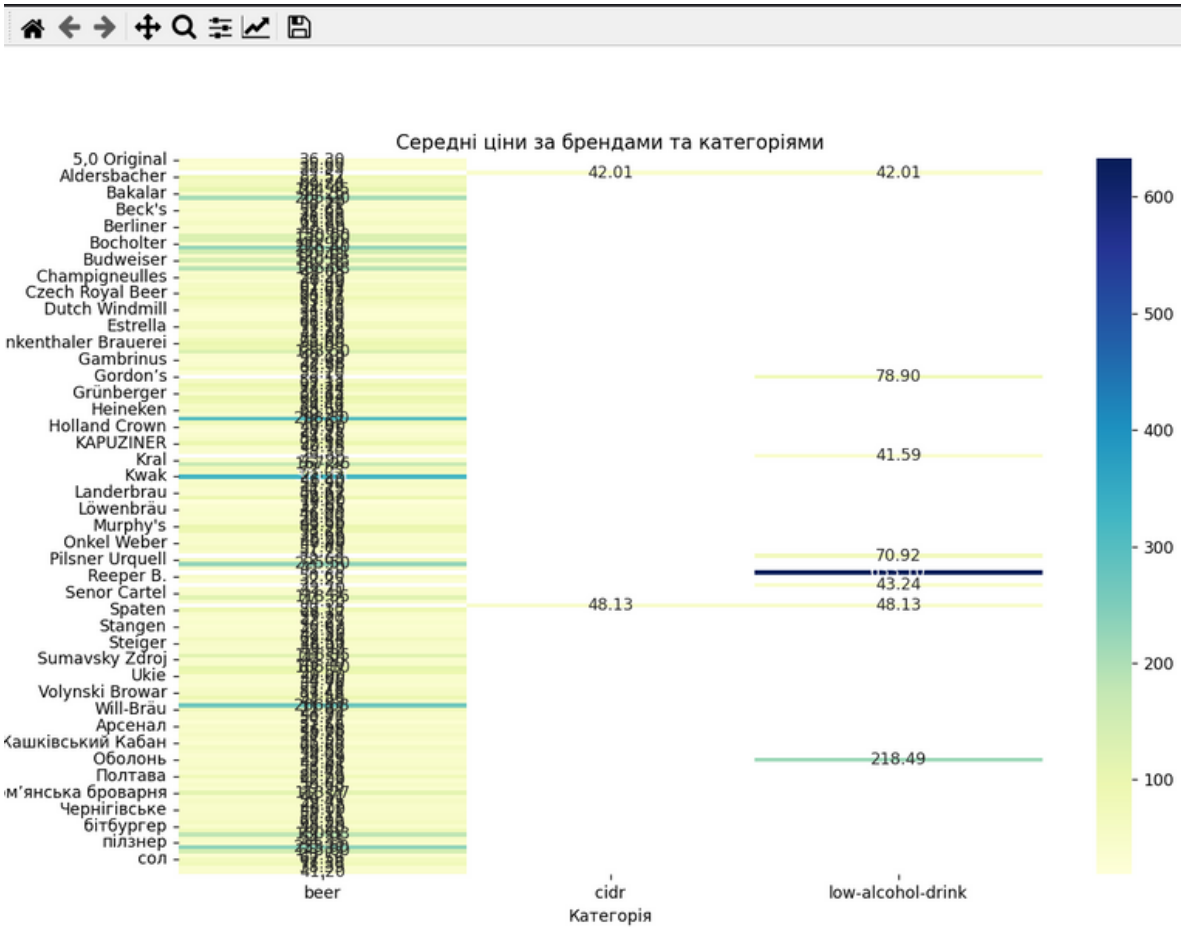

Zakaz.ua Parser
Парсинг по всім мережам
Вибірковий парсинг
Аналіз
Налаштування
Metro
МегаМаркет
WINETIME
METRO
ХАРКІВ
VARUS
ТАВРІЯ В
ЕКО МАРКЕТ
КОСМОС
NOVUS
ULTRAMARKET
восторг
Auchan
Алкогольні Напої
Сидр
Пиво
Слабоалкогольні напої
Напої
Солодка вода та напої
Мінеральна вода
Енергетики
Парсинг

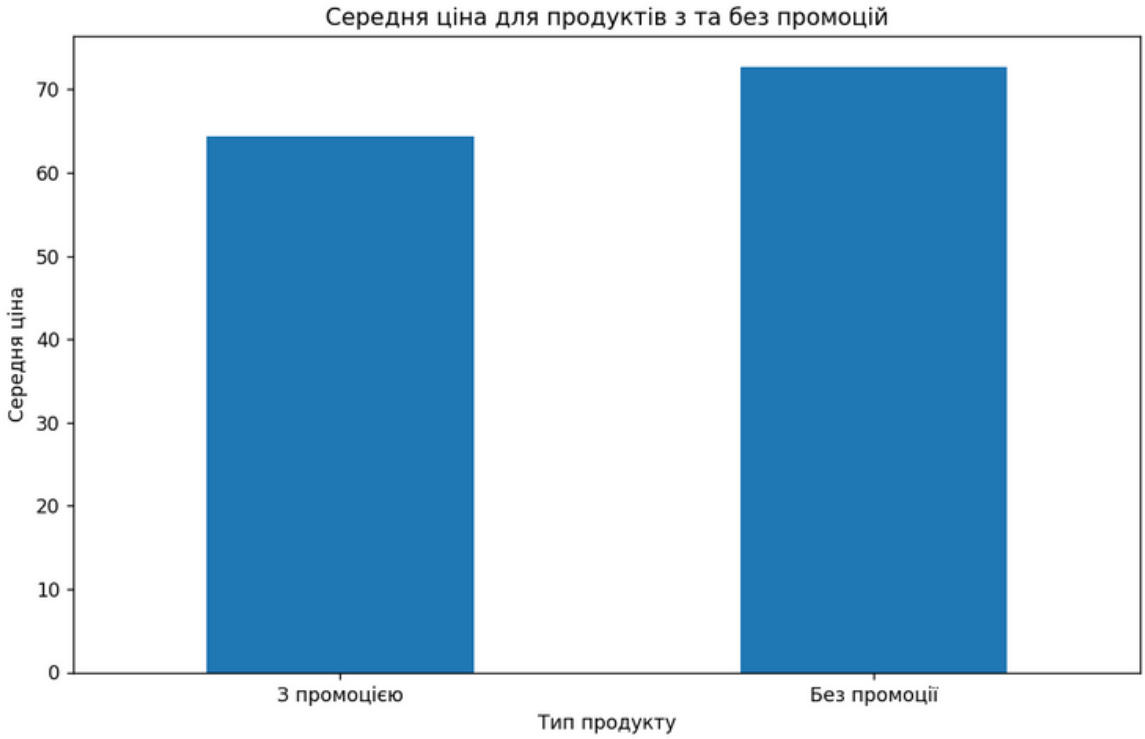

Zakaz.ua Parser
Парсинг по всім мережам
Вибірковий парсинг
Аналіз
Налаштування
Співвідношення ціна/обсяг
Запустити аналіз

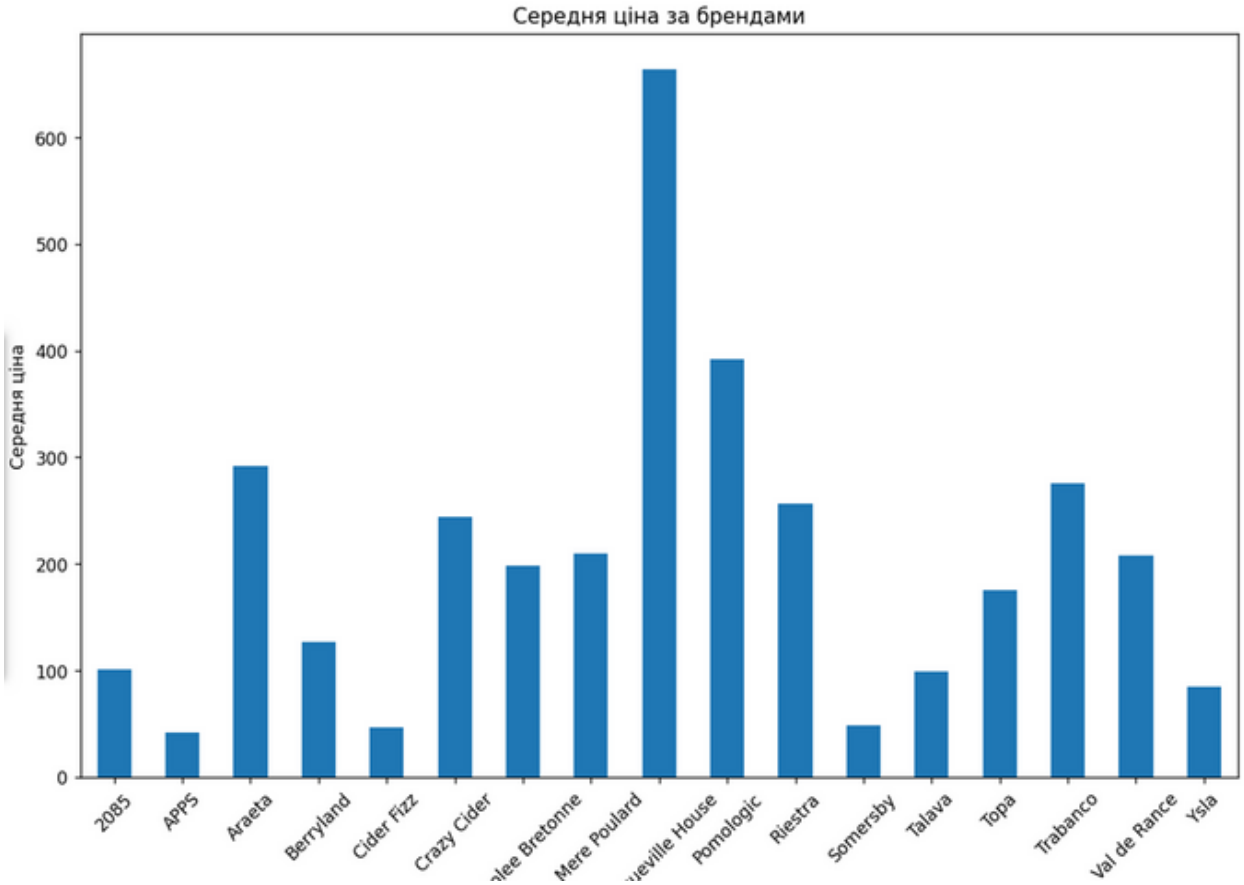

Zakaz.ua Parser
Парсинг по всім мережам
Вибірковий парсинг
Аналіз
Налаштування
Автозапуск парсера (Windows only)
Час запуску парсера: 0:00
Автопарсити кожні: 0
дні
Парсинг по всім мережам
Вибірковий парсинг
(Використовуються останні налаштування додатку)
Підтвердити

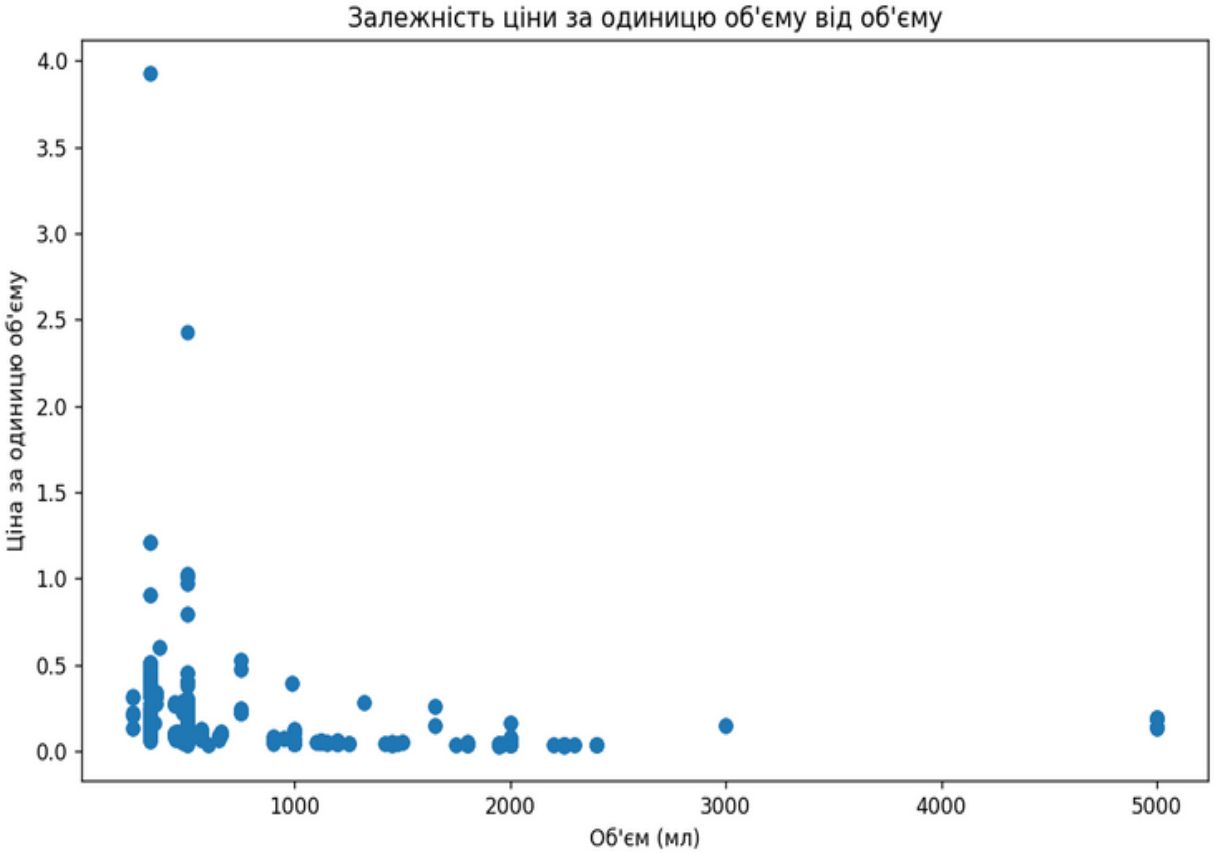
| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 04820097 | metro | low-alcoh | Напій слаб | 500мл | 1215.00 | | | 22.01.202 | REVO | https://metro.zakaz.ua |
| 04820097 | metro | low-alcoh | Напій слаб | 500мл | 47.84 | | | 22.01.202 | SHAKE | https://metro.zakaz.ua |
| 04820252 | metro | low-alcoh | Напій слаб | 500мл | 46.77 | | | 22.01.202 | SHAKE | https://metro.zakaz.ua |
| 04820000 | metro | low-alcoh | Напій Обс | 330мл | 34.79 | | | 22.01.202 | Оболонь | https://metro.zakaz.ua |
| 14820000 | metro | low-alcoh | Напій Обс | 330мл | 399.49 | | | 22.01.202 | Оболонь | https://metro.zakaz.ua |
| 14820000 | metro | low-alcoh | Напій Обс | 330мл | 399.49 | | | 22.01.202 | Оболонь | https://metro.zakaz.ua |
| 14820000 | metro | low-alcoh | Напій Обс | 330мл | 399.49 | | | 22.01.202 | Оболонь | https://metro.zakaz.ua |
| 14820029 | metro | low-alcoh | Напій Обс | 330мл | 399.49 | | | 22.01.202 | Оболонь | https://metro.zakaz.ua |
| 04820000 | metro | low-alcoh | Напій Обс | 330мл | 34.79 | | | 22.01.202 | Оболонь | https://metro.zakaz.ua |
| 04820000 | metro | low-alcoh | Напій Обс | 500мл | 45.58 | | | 22.01.202 | Оболонь | https://metro.zakaz.ua |
| 00000075 | megamark | beer | Пиво Corc | 330мл | 67.80 | | | 22.01.202 | Corona Ex | https://megamarket.za |
| 04820046 | megamark | beer | Пиво Krus | 500мл | 40.00 | 30.90 | 31.01.202 | 22.01.202 | Krusovice | https://megamarket.za |
| 04820000 | megamark | beer | Пиво Kron | 330мл | 115.60 | 85.60 | 31.01.202 | 22.01.202 | Kronenbou | https://megamarket.za |
| 04820046 | megamark | beer | Пиво Hein | 500мл | 203.00 | | | 22.01.202 | Heineken | https://megamarket.za |
| 04820000 | megamark | beer | Пиво Kron | 500мл | 45.60 | | | 22.01.202 | Kronenbou | https://megamarket.za |
| 08594404 | megamark | beer | Пиво Pilsn | 500мл | 62.20 | | | 22.01.202 | Pilsner Urc | https://megamarket.za |
| 04820252 | megamark | beer | Пиво Hais | 500мл | 35.00 | 26.60 | 23.01.202 | 22.01.202 | Haisenber | https://megamarket.za |
| 04820250 | megamark | beer | Пиво Льві | 480мл | 30.90 | | | 22.01.202 | Львівське | https://megamarket.za |
| 04820250 | megamark | beer | Пиво Льві | 1120мл | 61.60 | | | 22.01.202 | Львівське | https://megamarket.za |

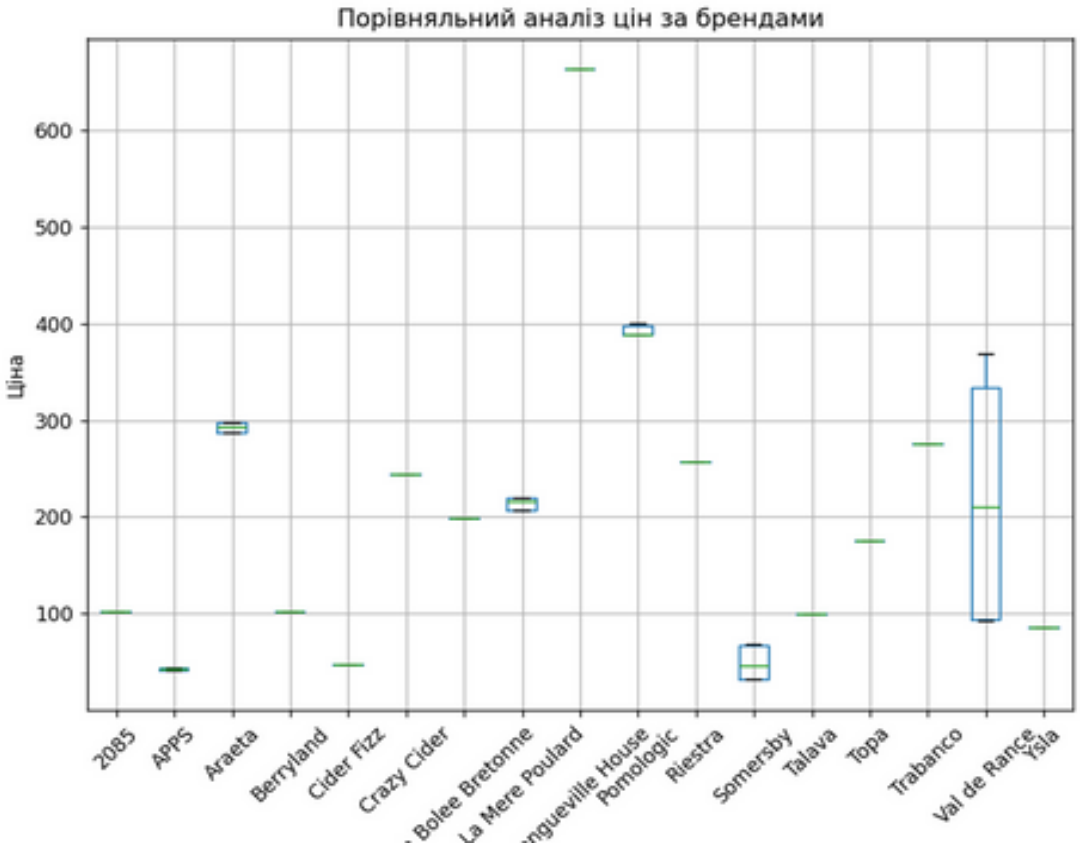



Середні ціни за брендами та категоріями
5,0 Original
Aldersbacher
Bakalar
Beck's
Berliner
Bocholter
Budweiser
Champigneulles
Czech Royal Beer
Dutch Windmill
Estrella
nkenthaler Brauerei
Gambrinus
Gordon's
Grünberger
Heineken
Holland Crown
KAPUZINER
Kral
Kwak
Landerbrau
Löwenbräu
Murphy's
Onkel Weber
Pilsner Urquell
Reeper B.
Senor Cartel
Spaten
Stangen
Steiger
Sumavsky Zdroj
Ukie
Volynski Browar
Will-Bräu
Арсенал
Кашківський Кабан
Оболонь
Полтава
м'янська броварня
Чернігівське
бітбургер
пілзнер
сол
42.01
42.01
78.90
41.59
70.92
43.24
48.13
48.13
218.49
600
500
400
300
200
100
beer
cidr
low-alcohol-drink
Категорія


Середня ціна для продуктів з та без промоцій
Середня ціна
70
60
50
40
30
20
10
0
З промоцією
Без промоції
Тип продукту


Середня ціна за брендами
Середня ціна
0
100
200
300
400
500
600
2085
APPS
Araeta
Berryland
Cider Fizz
Crazy Cider
Mere Poulard
Pomologic
Riestra
Somersby
Talava
Topa
Trabanco
Val de Rance
Ysla


Залежність ціни за одиницю об'єму від об'єму
Ціна за одиницю об'єму
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
1000
2000
3000
4000
5000
Об'єм (мл)


Порівняльний аналіз цін за брендами
Ціна
600
500
400
300
200
100
2085
APPS
Araeta
Berryland
Cider Fizz
Crazy Cider
Bolee Bretonne
La Mere Poulard
Pomologic
Riestra
Somersby
Talava
Topa
Trabanco
Val de Rance
Ysia

## Додаток Б

### Main.py

```
import ast
import parsermodule
from ZakazuaParser import Ui_MainWindow
import sys
import platform
import os
from configparser import ConfigParser
from  PyQt6.QtWidgets  import  QMainWindow,  QApplication,
QCheckBox, QPushButton, QComboBox
from ParserSettings import ParsingSettings
from       ParserWorkers       import       ParseAllWorker,
SelectiveParsingWorker
import resources_rc
import pandas as pd

import matplotlib
matplotlib.use('Qt5Agg')
import matplotlib.pyplot as plt
import seaborn as sns

config_file = "config.ini"

class MainWindow(QMainWindow):
    ui = None
    selected_shop_index = 0
    shop_list   =   ["megamarket",   "winetime",   "metro",
"kharkiv", "varus", "tavriav", "eko", "cosmos", "novus",
                "ultramarket", "vostorg", "auchan"]
    selected_parse_all_categories = list()

    def closeEvent(self, event):
        self.update_application_config()
        event.accept()

    def __init__(self):
        super(MainWindow, self).__init__()
        self.ui = Ui_MainWindow()
        self.ui.setupUi(self)
        self.parse_settings_list                              =
[ParsingSettings(self.shop_list[i]) for i in range(12)]

        length = len(self.parse_settings_list)
        for i in range(length):
```

```
            self.parse_settings_list[i].store              =
self.shop_list[i]
            self.parse_settings_list[i].categories = list()

        self.apply_application_config()

########################[Menu]########################
        self.ui.pbtn_menu_all_parsing.clicked.connect(
            lambda:
self.ui.stackedWidget.setCurrentWidget(self.ui.page_all_par
sing))
        self.ui.pbtn_menu_choose_parsing.clicked.connect(
            lambda:
self.ui.stackedWidget.setCurrentWidget(self.ui.page_choose_
parsing))
        self.ui.pbtn_menu_settings.clicked.connect(
            lambda:
self.ui.stackedWidget.setCurrentWidget(self.ui.page_setting
s))
        self.ui.pbtn_menu_analysis.clicked.connect(
            lambda:
self.ui.stackedWidget.setCurrentWidget(self.ui.page_analysi
s))

#####################################################


########################[Settings]####################
        self.ui.cbox_auto_start.clicked.connect(lambda:
self.check_auto_settings())

self.ui.pbtn_auto_start_apply.clicked.connect(lambda:
self.create_auto_task())

        if platform.system() != "Windows":
            self.ui.cbox_auto_start.setEnabled(False)

        if self.ui.cbox_auto_start.isChecked():
            self.ui.frame_time_interval.show()
        else:
            self.ui.frame_time_interval.hide()

#####################################################
        ####################[Parse
All]#####################
        self.ui.cbox_beer.clicked.connect(lambda:
```

```
self.set_category_for_parse_all("cidr", self.ui.cbox_beer))
        self.ui.cbox_cider.clicked.connect(lambda:
self.set_category_for_parse_all("cidr",
self.ui.cbox_cider))
        self.ui.cbox_mineral_water.clicked.connect(
            lambda:
self.set_category_for_parse_all("mineral-water",
self.ui.cbox_mineral_water))
        self.ui.cbox_energy_drinks.clicked.connect(
            lambda:
self.set_category_for_parse_all("energy-drinks",
self.ui.cbox_energy_drinks))
        self.ui.cbox_low_alcohol_drinks.clicked.connect(
            lambda:   self.set_category_for_parse_all("low-
alcohol-drink", self.ui.cbox_low_alcohol_drinks))

self.ui.cbox_sweet_water_and_drinks.clicked.connect(
            lambda:  self.set_category_for_parse_all("soft-
drinks", self.ui.cbox_sweet_water_and_drinks))

        self.ui.pbtn_parse_all.clicked.connect(lambda:
self.start_parsing_all())
        self.ui.pbar_parse_all.hide()

########################################################
        ###################[Selective
parsing]################

self.ui.pbtn_selective_parsing.clicked.connect(lambda:
self.start_selective_parsing())
        self.ui.pbar_selective_parsing.hide()

######################[Categories]####################
        self.ui.cbox_choose_cider.clicked.connect(
            lambda:
self.set_category_for_selected_shop("cidr",
self.ui.cbox_choose_cider))
        self.ui.cbox_choose_beer.clicked.connect(
            lambda:
self.set_category_for_selected_shop("beer",
self.ui.cbox_choose_beer))
        self.ui.cbox_choose_energy_drinks.clicked.connect(
            lambda:
self.set_category_for_selected_shop("energy-drinks",
self.ui.cbox_choose_energy_drinks))
        self.ui.cbox_choose_mineral_water.clicked.connect(
```

```
            lambda:
self.set_category_for_selected_shop("mineral-water",
self.ui.cbox_choose_mineral_water))

self.ui.cbox_choose_low_alcohol_drinks.clicked.connect(
            lambda:
self.set_category_for_selected_shop("low-alcohol-drink",
self.ui.cbox_choose_low_alcohol_drinks))

self.ui.cbox_choose_sweet_water_and_drinks.clicked.connect(
            lambda:
self.set_category_for_selected_shop("soft-drinks",
self.ui.cbox_choose_sweet_water_and_drinks))

##################################################
        ######################[Shop
Buttons]##################
        self.ui.pbtn_choose_auchan.clicked.connect(lambda:
self.update_categories_selected_shop("auchan",
self.ui.pbtn_choose_auchan))
        self.ui.pbtn_choose_eko.clicked.connect(lambda:
self.update_categories_selected_shop("eko",
self.ui.pbtn_choose_eko))
        self.ui.pbtn_choose_metro.clicked.connect(lambda:
self.update_categories_selected_shop("metro",
self.ui.pbtn_choose_metro))
        self.ui.pbtn_choose_novus.clicked.connect(lambda:
self.update_categories_selected_shop("novus",
self.ui.pbtn_choose_novus))
        self.ui.pbtn_choose_varus.clicked.connect(lambda:
self.update_categories_selected_shop("varus",
self.ui.pbtn_choose_varus))
        self.ui.pbtn_choose_cosmos.clicked.connect(lambda:
self.update_categories_selected_shop("cosmos",
self.ui.pbtn_choose_cosmos))
        self.ui.pbtn_choose_kharkiv.clicked.connect(lambda:
self.update_categories_selected_shop("kharkiv",
self.ui.pbtn_choose_kharkiv))

self.ui.pbtn_choose_megamarket.clicked.connect(lambda:
self.update_categories_selected_shop("megamarket",
self.ui.pbtn_choose_megamarket))
        self.ui.pbtn_choose_tavriav.clicked.connect(lambda:
self.update_categories_selected_shop("tavriav",
self.ui.pbtn_choose_tavriav))
        self.ui.pbtn_choose_vostorg.clicked.connect(lambda:
```

```
self.update_categories_selected_shop("vostorg",
self.ui.pbtn_choose_vostorg))

self.ui.pbtn_choose_ultramarket.clicked.connect(lambda:
self.update_categories_selected_shop("ultramarket",
self.ui.pbtn_choose_ultramarket))

self.ui.pbtn_choose_winetime.clicked.connect(lambda:
self.update_categories_selected_shop("winetime",
self.ui.pbtn_choose_winetime))

##################################################
        ######################[Analysis]#################

self.ui.run_analysis_button.clicked.connect(self.run_select
ed_analysis)

    @staticmethod
    def avrg_brand_category(df):
        average_prices        =        df.groupby(['brand',
'category'])['price'].mean().reset_index()
        # Перетворення результатів в широку форму для легшої
візуалізації
        pivot_table                                          =
average_prices.pivot_table(index='brand',
columns='category', values='price')
        # Візуалізація результатів
        plt.figure(figsize=(12, 8))
        sns.heatmap(pivot_table,   annot=True,   fmt=".2f",
cmap="YlGnBu")
        plt.title('Середні ціни за брендами та категоріями')
        plt.xlabel('Категорія')
        plt.ylabel('Бренд')
        plt.show()

    @staticmethod
    def avrg_price_promo(df):
        products_with_promo = df[df['promo'].notna()]
        products_without_promo = df[df['promo'].isna()]
        # Обчислення середніх цін для кожної групи
        average_price_with_promo                             =
products_with_promo['price'].mean()
        average_price_without_promo                          =
products_without_promo['price'].mean()
        # Виведення результатів
        print(f"Кількість    продуктів    з    промоціями:
```

```
{len(products_with_promo)}")
        print(f"Кількість продуктів без промоцій:
{len(products_without_promo)}")
        print(f"Середня ціна продуктів з промоціями:
{average_price_with_promo}")
        print(f"Середня ціна продуктів без промоцій:
{average_price_without_promo}")
        # Візуалізація даних
        promo_effect = pd.Series({'З промоцією':
average_price_with_promo, 'Без промоції':
average_price_without_promo})
        plt.figure(figsize=(10, 6))
        promo_effect.plot(kind='bar')
        plt.title('Середня ціна для продуктів з та без
промоцій')
        plt.xlabel('Тип продукту')
        plt.ylabel('Середня ціна')
        plt.xticks(rotation=0)
        plt.show()
    @staticmethod
    def analyze_brand_prices(df):
        print("Running selected analysis 3")
        unique_brands = df['brand'].unique()
        num_unique_brands = len(unique_brands)
        print("Кількість унікальних брендів:",
num_unique_brands)
        avg_prices_by_brand =
df.groupby('brand')['price'].mean()
        print("Середня ціна для кожного бренду:")
        print(avg_prices_by_brand)
        # Візуалізація даних
        plt.figure(figsize=(12, 6))
        avg_prices_by_brand.plot(kind='bar', rot=45)
        plt.title('Середня ціна за брендами')
        plt.xlabel('Бренд')
        plt.ylabel('Середня ціна')
        plt.ion()
        plt.show()

    @staticmethod
    def price_vol(df):
        sns.set(style="darkgrid")
        plt.figure(figsize=(10, 6))
        df['vol'] =
df['vol'].str.extract('(\d+)').astype(float)
        sns.scatterplot(x='vol', y='price', data=df)
```

```
        sns.regplot(x='vol',      y='price',      data=df,
scatter=False, color='red')
        plt.title('Співвідношення ціна/обсяг')
        plt.xlabel('Обсяг продукції')
        plt.ylabel('Ціна')
        plt.show()

    @staticmethod
    def analyze_brand(df):
        df.boxplot(column='price',              by='brand',
showfliers=False)
        plt.title('Порівняльний аналіз цін за брендами')
        plt.xlabel('Бренд')
        plt.ylabel('Ціна')
        plt.xticks(rotation=45)
        plt.ion()
        plt.show()

    @staticmethod
    def price_per_vol(df):
        df['vol']       =       df['vol'].str.replace('мл',
'').astype(float)
        df['price_per_vol'] = df['price'] / df['vol']
        plt.figure(figsize=(10, 6))
        plt.scatter(df['vol'], df['price_per_vol'])
        plt.title("Залежність ціни за одиницю об'єму від
об'єму")
        plt.xlabel("Об'єм (мл)")
        plt.ylabel("Ціна за одиницю об'єму")
        plt.show()


    def run_selected_analysis(self):
        print("Running selected analysis")
        current_directory = os.path.dirname(sys.executable)
        file_path     =     os.path.join(current_directory,
'zakaz.xlsx')
        df = pd.read_excel(file_path)
        selected_method                                    =
self.ui.analysis_method_combo.currentText()
        print('123+' + selected_method)
        if selected_method == "Співвідношення ціна/обсяг":
            self.price_vol(df)
        elif selected_method == "Середня ціна за брендами":
            self.analyze_brand_prices(df)
        elif selected_method == "Порівняльний аналіз цін за
```

```
брендами":
            self.analyze_brand(df)
        elif selected_method == "Середня ціна для продуктів
з та без промоцій":
            self.avrg_price_promo(df)
        elif selected_method == "Середні ціни за брендами та
категоріями":
            self.avrg_brand_category(df)
        elif selected_method == "Залежність ціни за одиницю
об'єму від об'єму":
            self.price_per_vol(df)




    def update_parse_all_progress_bar(self, n):
        self.ui.pbar_parse_all.show()
        self.ui.pbar_parse_all.setValue(n)

    def update_selective_parse_progress_bar(self, n):
        self.ui.pbar_selective_parsing.show()
        self.ui.pbar_selective_parsing.setValue(n)

    def disable_parse_buttons(self):
        self.ui.pbtn_parse_all.setEnabled(False)
        self.ui.pbtn_selective_parsing.setEnabled(False)

    def enable_parse_buttons(self):
        self.ui.pbtn_parse_all.setEnabled(True)
        self.ui.pbtn_selective_parsing.setEnabled(True)

    def               update_selected_shop_stylesheet(self,
selected_shop=QPushButton):
        shop_buttons                                        =
self.ui.frame_shop_buttons.findChildren(QPushButton)
        for button in shop_buttons:
            if isinstance(button, QPushButton):
                button.setStyleSheet("")
        if selected_shop is not None:
            selected_shop.setStyleSheet("background-color:
rgb(54, 123, 245);")

    def  set_category_for_parse_all(self,  category=str(),
check_box=QCheckBox):
        if check_box.isChecked():
```

```
self.selected_parse_all_categories.append(category)
        else:

self.selected_parse_all_categories.remove(category)

    def                set_category_for_selected_shop(self,
category=str(), check_box=QCheckBox):
        if check_box.isChecked():

self.parse_settings_list[self.selected_shop_index].categori
es.append(category)
        else:

self.parse_settings_list[self.selected_shop_index].categori
es.remove(category)

    def  update_categories_selected_shop(self,  shop_name,
selected_shop):
        self.uncheck_all_category_checkboxes()

self.ui.label_selective_parsing_shop_name.setText(shop_name
.capitalize())
        self.selected_shop_index                          =
self.shop_list.index(shop_name)
        self.update_selected_shop_stylesheet(selected_shop)

        CheckBoxes = {"cidr": self.ui.cbox_choose_cider,
                      "beer": self.ui.cbox_choose_beer,
                      "energy-drinks":
self.ui.cbox_choose_energy_drinks,
                      "mineral-water":
self.ui.cbox_choose_mineral_water,
                      "low-alcohol-drink":
self.ui.cbox_choose_low_alcohol_drinks,
                      "soft-drinks":
self.ui.cbox_choose_sweet_water_and_drinks
                      }

        categories                                        =
self.parse_settings_list[self.selected_shop_index].categori
es
        categories_amount = len(categories)
        for i in range(categories_amount):
            CheckBoxes.get(categories[i]).setChecked(True)

    def uncheck_all_category_checkboxes(self):
```

```
        categories                                          =
self.ui.page_choose_parsing.findChildren(QCheckBox)
        for category in categories:
            if isinstance(category, QCheckBox):
                category.setChecked(False)

    def check_auto_settings(self):
        if self.ui.cbox_auto_start.isChecked():
            self.ui.frame_time_interval.show()
        else:
            self.ui.frame_time_interval.hide()
            self.delete_auto_task()

    def create_auto_task(self):
        if platform.system() == "Windows":
            task = f'schtasks /create /tn "ZakazuaParser"
/tr   "\'{str(sys.executable)}\'   -ap"   /sc   daily   /mo
{str(self.ui.time_interval_in_days.value())}            /st
{self.ui.time_interval.time().toString("HH:mm")} /f'
            os.system('chcp 65001')
            os.system(task)

    def delete_auto_task(self):
        if platform.system() == "Windows":
            task = f'schtasks /delete /tn "ZakazuaParser"
/f'
            os.system('chcp 65001')
            os.system(task)

    def update_application_config(self):
        config = ConfigParser()
        config.add_section('Settings')
        config.set('Settings',   'Is_Auto_Parsing_Enabled',
str(self.ui.cbox_auto_start.isChecked()))
        config.set('Settings',            'Auto_Parse_All',
str(self.ui.rbtn_settings_parse_all.isChecked()))
        config.set('Settings',      'Auto_Selective_Parse',
str(self.ui.rbtn_settings_selective_parsing.isChecked()))
        config.add_section('Parse_All')
        config.set('Parse_All',              'categories',
str(self.selected_parse_all_categories))
        config.add_section('Selective_Parsing')
        for shop_settings in self.parse_settings_list:
            config.set('Selective_Parsing',
shop_settings.store, str(shop_settings.categories))
        with open(config_file, 'w+') as configfile:
```

```
            config.write(configfile)

    def apply_application_config(self):
        config = ConfigParser()
        if os.path.isfile(config_file):
            config.read(config_file)

self.ui.cbox_auto_start.setChecked(config.getboolean('Setti
ngs', 'Is_Auto_Parsing_Enabled', fallback=False))

self.ui.rbtn_settings_parse_all.setChecked(config.getboolea
n('Settings', 'Auto_Parse_All', fallback=False))

self.ui.rbtn_settings_selective_parsing.setChecked(config.g
etboolean('Settings',              'Auto_Selective_Parse',
fallback=False))
            CheckBoxes = {"cidr": self.ui.cbox_cider,
                          "beer": self.ui.cbox_beer,
                          "energy-drinks":
self.ui.cbox_energy_drinks,
                          "mineral-water":
self.ui.cbox_mineral_water,
                          "low-alcohol-drink":
self.ui.cbox_low_alcohol_drinks,
                          "soft-drinks":
self.ui.cbox_sweet_water_and_drinks
                          }
            self.selected_parse_all_categories            =
ast.literal_eval(config['Parse_All']['categories'])
            for                category                in
self.selected_parse_all_categories:
                CheckBoxes.get(category).setChecked(True)

            parse_settings                                =
dict(config.items('Selective_Parsing'))
            for shop_settings in self.parse_settings_list:
                shop_settings.categories                  =
ast.literal_eval(parse_settings.get(shop_settings.store))

self.update_categories_selected_shop(self.shop_list[self.se
lected_shop_index], None)

    def start_parsing_all(self):
        self.parsing_all_worker                           =
ParseAllWorker(self.shop_list[:],
self.selected_parse_all_categories[:])
```

```
self.parsing_all_worker.progress.connect(self.update_parse_
all_progress_bar)

self.parsing_all_worker.started.connect(self.disable_parse_
buttons)

self.parsing_all_worker.finished.connect(self.enable_parse_
buttons)
        self.parsing_all_worker.start()

    def start_selective_parsing(self):
        self.selective_parser_worker                          =
SelectiveParsingWorker(self.parse_settings_list[:])

self.selective_parser_worker.progress.connect(self.update_s
elective_parse_progress_bar)

self.selective_parser_worker.started.connect(self.disable_p
arse_buttons)

self.selective_parser_worker.finished.connect(self.enable_p
arse_buttons)
        self.selective_parser_worker.start()

def get_parse_all_category_list():
    category_list = None
    if os.path.isfile(config_file):
        config = ConfigParser()
        config.read(config_file)
        category_list = config['Parse_All']['categories']
        category_list = ast.literal_eval(category_list)
    return category_list


def get_selective_parse_settings_list():
    parse_settings = None
    if os.path.isfile(config_file):
        config = ConfigParser()
        config.read(config_file)
        parse_settings                                        =
dict(config.items('Selective_Parsing'))
        for shop in parse_settings:
            parse_settings[shop]                              =
ast.literal_eval(parse_settings[shop])
    return parse_settings
```

```
def start_parsing_all():
    parsing = parsermodule.Client()
    categories = get_parse_all_category_list()
    if categories is not None:
        parsing.run(categories, parsermodule.shop_list)
    sys.exit(0)


def start_selective_parsing():
    parsing = parsermodule.Client()
    parse_settings = get_selective_parse_settings_list()
    if parse_settings is not None:
        for shop in parse_settings:
            for category in parse_settings[shop]:
                parsing.run(category, shop)
    sys.exit(0)


def check_application_arguments(args):
    for arg in args:
        if arg == "-ap":
            if os.path.isfile(config_file):
                config = ConfigParser()
                config.read(config_file)
                if            config.getboolean('Settings',
'Auto_Parse_All'):
                    start_parsing_all()
                elif          config.getboolean('Settings',
'Auto_Selective_Parse'):
                    start_selective_parsing()


if __name__ == "__main__":
    check_application_arguments(sys.argv)
    app = QApplication(sys.argv)
    window = MainWindow()
    window.show()
    sys.exit(app.exec())
```

**parsermodule.py**

```
import sys
from datetime import datetime
import logging
import collections
import time
import re
import bs4
import requests
import os
import openpyxl

logging.basicConfig(level=logging.DEBUG)
logger = logging.getLogger('zakaz.ua')

ParserResult = collections.namedtuple(
    'ParserResult',
    (
        'product_key',
        'brand_name',
        'goods_name',
        'url',
        'price',
        'price_discount',
        'expiration_date',
        'volume',
    )
)

HEADERS = (
    'num',
    'retail',
    'category',
    'sku',
    'vol',
    'price',
    'promo',
    'promo date',
    'date',
    'brand',
    'url',
)

shop_list = ["megamarket", "winetime", "metro", "kharkiv",
"varus", "eko", "stolychnyi", "cosmos", "novus",
             "ultramarket", "vostorg", "auchan"]
```

```
alcohol_drinks = ["beer", "cidr", "low-alcohol-drink"]
beverages = ["mineral-water", "soft-drinks", "energy-
drinks"]
class Client:

    def __init__(self):
        self.session = requests.Session()
        self.session.headers = {
            "Accept":"*/*",
            "User-Agent": "Mozilla/5.0 (Windows NT 10.0;
Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/117.0.0.0 Safari/537.36",
            "Accept-Language": "ru,uk-UA;q=0.9,uk;q=0.8,en-
",
        }
        self.result = []

    def get_total_pages(self, text):
        soup = bs4.BeautifulSoup(text, 'lxml')
        pagination = soup.find('ul', class_='Pagination')
        if pagination:
            total_pages = soup.find_all('a',
class_="Pagination__item")[-1].get_text()
            logger.info(f'Сторінок {total_pages}')
            return total_pages
        return 1

    def load_page(self, product_category, shop, page: int =
1):
        if shop == "eko":
            base_url =
f"https://{shop}.zakaz.ua/uk/categories/{product_category}-
{shop}market/available=1/?page={page}"
        elif shop == "megamarket" and product_category ==
"low-alcohol-drink":
            base_url =
f"https://{shop}.zakaz.ua/uk/categories/low-alcohol-
{shop}/available=1/?page={page}"
        else:
            base_url =
f"https://{shop}.zakaz.ua/uk/categories/{product_category}-
{shop}/available=1/?page={page}"
        url = base_url.format(page)
        res = self.session.get(url=url)
        #time.sleep(1)
```

```
        res.raise_for_status()
        return res.text

    def parse_page(self, text: str, shop):
        soup = bs4.BeautifulSoup(text, 'lxml')
        products   =   soup.find_all('a',   {'data-testid':
'product-tile'})
        for product in products:
            product_block   =   product.find_parent('div',
class_='ProductsBox__listItem')
            self.parse_block(product_block, shop)
        logger.debug('Сторінку пропарсено!')


    def parse_block(self,block,shop):
        product_tile      =      block.select_one('a[data-
testid="product-tile"]')
        if product_tile:
            product_key      =      product_tile.get('data-
productkey')
            url_block      =      block.select_one('a[data-
testid="product-tile"]')
            url = url_block.get('href')
            logger.debug('Код товару: %s', product_key)
            full_url = f"https://{shop}.zakaz.ua{url}"

            def get_page_content(url):
                response = requests.get(url)
                response.raise_for_status()
                time.sleep(1)
                return response.text

            def parse_product_details(html_content):

                def extract_expiration_date(text):
                    date_pattern = r'(\d{2}\.\d{2}\.\d{4})'
                    matches   =   re.findall(date_pattern,
text)
                    for match in matches:
                        try:
                            datetime.strptime(match,
'%d.%m.%Y')
                            return match
                        except ValueError:
                            continue
                    return None
```

```
                soup    =    bs4.BeautifulSoup(html_content,
'lxml')

                product_name   =   soup.find('h1',   {'data-
marker': 'Big Product Cart Title'}).text.strip()
                logger.debug('Назва: %s', product_name)

                expiration_date = None
                discount_value = None
                discount   =   soup.find('div',   {'class':
'BigProductCardTopInfo__discountDisclaimer'})
                if discount:
                    price   =   soup.find('span',   {'data-
marker-value': 'Price'})
                    price_value = price.text.strip()
                    logger.debug('Ціна: %s', price_value)

                    discount = soup.find('span', {
                        'data-marker-value':         'Price
Discount'})
                    discount_value = discount.text.strip()
                    expiration_date_text                  =
soup.find('span', {
                                'class':
'DiscountDisclaimer_subhead'}).text.strip()
                    logger.debug('expiration_date_text:
%s', expiration_date_text)
                    expiration_date                       =
extract_expiration_date(expiration_date_text)            if
expiration_date_text else None
                    logger.debug('Строк  дії  знижки:  %s',
expiration_date)
                else:
                    price   =   soup.find('span',   {'data-
marker-value': 'Price'})
                    price_value = price.text.strip()
                    logger.debug('Ціна: %s', price_value)

                brand = soup.find('span', {'class': 'jsx-
118055876e04a4fd   BigProductCardDescription__entryTitle'},
string='Бренд')
                brand_name = brand.find_next('span', {
                    'class':          'jsx-118055876e04a4fd
BigProductCardDescription__entryValue'}).text.strip()    if
brand else None
```

```
                logger.debug('Бренд: %s', brand_name)

                volume = soup.find('span', {'class': 'jsx-
118055876e04a4fd   BigProductCardDescription__entryTitle'},
string='Об\'єм')
                volume_value = volume.find_next('span', {
                    'class':          'jsx-118055876e04a4fd
BigProductCardDescription__entryValue'}).text.strip()    if
volume else None
                logger.debug('Об\'єм: %s', volume_value)

                return      product_name,      price_value,
discount_value, brand_name, volume_value, expiration_date

            html_content = get_page_content(full_url)
            product_name, price, discount, brand, volume,
expiration_date = parse_product_details(html_content)

            self.result.append(ParserResult(
                product_key=product_key,
                brand_name=brand,
                goods_name=product_name,
                url=full_url,
                price=price,
                price_discount=discount,
                expiration_date=expiration_date,
                volume=volume,
            ))
            if discount:
                logger.debug('Знижка: %s грн', discount)
            logger.debug('URL товару: %s', full_url)


    def save_result(self, category, shop):
        current_directory = os.path.dirname(sys.executable)
        file_path     =     os.path.join(current_directory,
'zakaz.xlsx')
        if not os.path.exists(file_path):
            wb = openpyxl.Workbook()
            ws = wb.active
            ws.append(HEADERS)
        else:
            wb = openpyxl.load_workbook(file_path)
            ws = wb.active

        for item in self.result:
```

```
            row = (item.product_key, shop, category,
item.goods_name, item.volume, item.price,
                   item.price_discount,
item.expiration_date,     datetime.now().strftime("%d.%m.%Y
%H:%M:%S"), item.brand_name, item.url)
            ws.append(row)

        wb.save(file_path)



    def run(self, product_category, shop):
        first_page_text = self.load_page(product_category,
shop)
        total_pages = self.get_total_pages(first_page_text)
        for page in range(1, int(total_pages) + 1):
            page_text = self.load_page(product_category,
shop=shop, page=page)
            self.parse_page(text=page_text, shop=shop)
            logger.info(f'Товарів на сторінці {page}:
{len(self.result)} шт.')
        logger.info(f'Усього товарів: {len(self.result)}
шт.')
        self.save_result(product_category, shop)



if __name__ == '__main__':
    parser = Client()
    product_categories = ['cidr']
    shops = ['metro']

    for product_category in product_categories:
        logger.info(f'Категорія {product_category}')
        for shop in shops:
            logger.info(f'Магазин {shop}')
            parser.run(product_category, shop)
```

**ParserSettings.py**

```
class ParsingSettings:
    def __init__(self, store):
        self.store = store
        self.categories = list()
```

**ParserWorkers.py**

```
from PyQt6.QtCore import QThread, pyqtSignal
import parsermodule
from ParserSettings import ParsingSettings


class ParseAllWorker(QThread):
    progress = pyqtSignal(int)

    def __init__(self, shop_list, selected_parse_all_categories, parent=None):
        super(QThread, self).__init__(parent=parent)
        self.shop_list = shop_list
        self.selected_parse_all_categories = selected_parse_all_categories
        self.total_number_of_categories = len(selected_parse_all_categories) * len(shop_list)

    def run(self):
        self.progress.emit(0)
        parser = parsermodule.Client()
        complete_category_amount = 0
        if len(self.selected_parse_all_categories) != 0:
            for shop in self.shop_list:
                for product_category in self.selected_parse_all_categories:
                    parser.run(product_category, shop)
                    complete_category_amount += 1

self.progress.emit(int((complete_category_amount/self.total_number_of_categories)*100))


class SelectiveParsingWorker(QThread):
    progress = pyqtSignal(int)

    def __init__(self, parse_settings_list, parent=None):
        super(QThread, self).__init__(parent=parent)
        self.parse_settings_list = parse_settings_list
```

```
        self.total_number_of_categories = 0
        for settings in self.parse_settings_list:
            for category in settings.categories:
                self.total_number_of_categories += 1

    def run(self):
        self.progress.emit(0)
        parsing = parsermodule.Client()
        if len(self.parse_settings_list) != 0:
            complete_category_amount = 0
            for settings in self.parse_settings_list:
                if isinstance(settings, ParsingSettings):
                    if len(settings.categories) != 0:
                        for           category           in
settings.categories:
                            parsing.run(category,
settings.store)
                            complete_category_amount += 1

self.progress.emit(int((complete_category_amount          /
self.total_number_of_categories) * 100))
```