Міністерство освіти і науки України
Університет митної справи та фінансів

Факультет інноваційних технологій
Кафедра комп'ютерних наук та інженерії програмного забезпечення

Кваліфікаційна робота бакалавра

на тему

«Розробка рекомендаційної системи для компаній в сфері роздрібної торгівлі з використанням методу інтелектуального аналізу даних»

Виконав: студент групи К21-2
Спеціальність 122 «Комп'ютерні науки»
Савчук Максим Анатолійович
Керівник_____
_____
Рецензент_____
_____
_____

Дніпро – 2025

АНОТАЦІЯ

Савчук М.А. Розробка рекомендаційної системи для компаній в сфері роздрібної торгівлі з використанням методу інтелектуального аналізу даних.

Кваліфікаційна робота на здобуття освітнього ступеня бакалавр за спеціальністю 122 «Комп'ютерні науки». – Університет митної справи та фінансів, Дніпро, 2025.

У кваліфікаційній роботі представлено розробку персоналізованої рекомендаційної системи для компаній у сфері роздрібної торгівлі. Було проаналізовано основні методи інтелектуального аналізу даних, що використовуються при розробці рекомендацій систем, зокрема детально було проаналізовано метод асоціативних правил. Реалізована система формує рекомендації на основі аналізу історії покупок інших користувачів та виявляє частотні шаблони спільного придбання товарів.

Використано метод пошуку асоціативних правил, а саме алгоритм Apriori , адаптований для умов реального магазину з базою даних транзакцій. У розробленій програмі користувач може додавати товари до кошика, після чого система формує персоналізовані рекомендації товарів, які найчастіше купуються разом із вибраними. Додатково реалізовано механізм перегляду частот появи комбінацій товарів в історії, що забезпечує прозорість роботи системи.

Ключові слова: рекомендаційна система, асоціативні правила, роздрібна торгівля, Apriori, аналіз даних, транзакції, C#, SQL, персоналізація.

ЗМІСТ

ВСТУП

Останні десятиліття позначені стрімким розвитком цифрових технологій, що докорінно трансформували підходи до ведення бізнесу в багатьох галузях. Однією з таких галузей стала роздрібна торгівля — сфера, де взаємодія із клієнтом є безпосередньою, постійною та критично залежною від точності комунікації, гнучкості сервісу та швидкості реагування на запити споживача. Із переходом значної частини комерційної активності в онлайн-середовище, компанії почали стикатися з новими викликами, головним серед яких стало питання ефективного використання інформації, що накопичується внаслідок транзакційної взаємодії з покупцем.

Сучасний покупець очікує не лише великого вибору товарів, а й того, що цей вибір буде зручним, швидким і, головне, персоналізованим. Людина, яка здійснює покупки в цифровому середовищі, розраховує на те, що система, з якою вона взаємодіє, буде враховувати її вподобання, звички, а іноді — й наміри, ще не висловлені в діях. Саме тому на передній план виходять рекомендаційні системи — інструменти, що на основі накопичених даних здатні формувати релевантні, а іноді й передбачувані пропозиції, що максимально відповідають інтересам конкретного користувача.

Попри очевидні переваги таких систем, їх впровадження у практику роздрібної торгівлі пов'язане з низкою труднощів. По-перше, не всі підприємства мають у своєму розпорядженні достатні технічні та людські ресурси для використання складних алгоритмів машинного навчання або розгортання хмарних аналітичних платформ. По-друге, навіть якщо дані доступні, їх обсяг або структура часто не дозволяють застосувати класичні методи персоналізації, що базуються на профілюванні користувачів чи глибокому семантичному аналізі товарів. У таких умовах виникає потреба в альтернативних підходах — простих у реалізації, зрозумілих для користувача та ефективних з точки зору бізнес-результату.

У межах цього дослідження було обрано саме такий підхід — побудову системи рекомендацій, що ґрунтується на транзакційній інформації, без необхідності використання персональних даних або складної моделі користувацьких вподобань. Основною ідеєю стало виявлення стійких зв'язків між товарами, що часто купуються разом, і подальше використання цих зв'язків для формування персоналізованих рекомендацій на основі поточного вмісту кошика користувача. Такий метод має низку переваг: він не вимагає попереднього навчання моделі, легко масштабовується в межах локального середовища та забезпечує прозорість прийняття рішень — кожна рекомендація може бути логічно обґрунтована частотою попередніх поєднань товарів.

В основі запропонованої системи лежить адаптація методу асоціативного аналізу, зокрема алгоритму Apriori, який у своєму класичному вигляді передбачає ітеративне виявлення частих множин елементів у великих обсягах даних. Проте з огляду на практичні обмеження було реалізовано спрощену модель, де акцент зміщено на перевірку повного входження поточного набору товарів у історичні транзакції з метою виявлення найбільш релевантних додаткових позицій. Цей підхід не лише значно знижує обчислювальну складність, а й дозволяє будувати систему, яка реагує в реальному часі на дії користувача.

Особливу увагу було приділено архітектурі майбутнього рішення. Оскільки метою було створення настільної програми для малих торгових підприємств, що не мають постійного доступу до інтернету або відмовляються від використання сторонніх хмарних сервісів, реалізація базувалася на платформі Windows Forms з використанням мови C# та локальної SQL-бази даних. Такий вибір забезпечив автономність роботи системи, простоту її розгортання та можливість інтеграції у вже існуючі процеси роздрібної торгівлі без потреби кардинальних змін у інфраструктурі.

Окремим завданням у межах дослідження стало проектування структури бази даних, яка дозволяє не лише зберігати інформацію про користувачів,

товари й замовлення, а й ефективно опрацьовувати транзакції для подальшого аналізу. Особливістю реалізації стало збереження замовлень у вигляді текстових рядків із переліком товарів і кількостей — такий підхід значно спрощує синтаксичну обробку на стороні клієнтської програми, дозволяючи уникнути складної SQL-аналітики.

У ході розробки системи було сформовано інтуїтивний графічний інтерфейс, що забезпечує комфортну взаємодію з користувачем. Усі дії — від авторизації до формування замовлення та перегляду рекомендацій — реалізовано з урахуванням принципів сучасного UX-дизайну. Програма реагує на дії користувача миттєво, а результати аналізу з'являються одразу після додавання товару до кошика. Крім того, реалізовано можливість перегляду частотних зв'язків між товарами, що дозволяє краще зрозуміти логіку роботи системи та підвищити довіру до рекомендацій.

Результати тестування продемонстрували, що запропонована система здатна ефективно працювати навіть за умов обмеженого набору транзакційних даних. У більшості випадків рекомендації відповідали логіці користувача, а простота реалізації дозволила швидко адаптувати програму до різних умов. Важливо й те, що система не залежить від конфіденційної інформації, що знімає низку юридичних і етичних ризиків, пов'язаних із персоналізацією.

Таким чином, дослідження охоплює не лише технічний аспект побудови рекомендаційної системи, а й питання її доцільності, релевантності до потреб малого бізнесу та відповідності сучасним тенденціям розвитку цифрової торгівлі. Особлива цінність полягає в тому, що підхід, який зазвичай асоціюється з великими платформами та складними аналітичними системами, було адаптовано до умов, у яких найчастіше працюють локальні магазини, міні-маркети та торгові точки. Це дозволяє розглядати результати як приклад демократизації інтелектуальних технологій у роздрібній торгівлі.

Побудована система не є універсальним вирішенням усіх викликів персоналізації, однак вона наочно демонструє, що за наявності якісно структурованих даних і чітко визначеної задачі, можна реалізувати ефективне

прикладне рішення без надмірної складності. Вона також відкриває можливості для подальшого розвитку — зокрема, інтеграції з мобільними застосунками, розширення бази транзакцій, упровадження більш складних алгоритмів кластеризації або фільтрації. У цьому сенсі дана робота не лише виконує роль завершеного інженерного проєкту, але й є підґрунтям для подальших досліджень і вдосконалень у сфері систем персоналізованих рекомендацій.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Особливості функціонування роздрібної торгівлі

Роздрібна торгівля є ключовим елементом економіки будь-якої країни, забезпечуючи безпосередній зв'язок між виробниками товарів та кінцевими споживачами. Вона включає різноманітні форми та канали продажів, починаючи від традиційних фізичних магазинів до електронної комерції, що продовжує зростати в умовах цифровізації. Сучасний стан роздрібної торгівлі характеризується високою конкуренцією, значними змінами в споживацьких уподобаннях, а також впливом новітніх технологій на процеси продажу та обслуговування клієнтів [1; 9].

Особливу роль у функціонуванні роздрібної торгівлі відіграють методи аналізу і прогнозування поведінки споживачів, що дає можливість компаніям створювати персоналізовані пропозиції для своїх клієнтів. В умовах швидких змін у вподобаннях покупців важливим є вміння швидко реагувати на зміни в попиті, а також ефективно використовувати зібрані дані для оптимізації маркетингових стратегій.

Розвиток IT-сфери створює нові перспективи для впровадження рекомендаційних систем. [8; 10; 24], які використовують різноманітні алгоритми для обробки та аналізу значних масивів даних та формування індивідуальних рекомендацій для користувачів. Такі системи здатні не лише покращити досвід покупців, але й сприяти зростанню доходів підприємств через підвищення лояльності та конверсії клієнтів. Водночас для ефективної роботи таких систем потребує врахування специфіка роздрібного бізнесу, що включає широкий спектр товарів, постійні зміни в асортименті, а також різноманітні маркетингові акції, що можуть впливати на вибір споживачів.

У контексті зазначеного, розробка рекомендаційних систем для компаній у сфері роздрібної торгівлі є актуальним завданням, яке передбачає необхідність адаптації алгоритмів до конкретних умов бізнесу. Тому

важливим є дослідження особливостей функціонування таких систем, що дозволить створити ефективні механізми для персоналізації пропозицій та підвищення їх релевантності для кінцевого споживача.

Динамічність ринку роздрібної торгівлі змушує компанії постійно переглядати підходи до взаємодії з клієнтом. На зміну масовому маркетингу приходять індивідуальні моделі комунікації, де кожна взаємодія формується спираючись на результати аналізу минулих дій користувача. Завдяки цьому виникає попит на системи, які здатні в реальному часі формувати пропозиції, релевантні до потреб конкретного споживача. І саме тут рекомендаційні системи виявляються не просто додатковим функціоналом, а інструментом стратегічного значення.

Слід також врахувати, що роздрібна торгівля — це не лише технології, але й люди. Вибір споживача завжди залишається багатофакторним явищем, що включає емоційні, соціальні, сезонні та ситуативні чинники. Завданням сучасних рекомендаційних систем є не просто врахувати історію покупок, а виявити закономірності, що можуть сигналізувати про приховані потреби або наміри клієнта. Для цього потрібно поєднувати алгоритмічну точність з гнучкістю моделей, здатних пристосовуватися до мінливого поведінкового контексту.

На додачу, необхідно зважати на особливості малого та середнього бізнесу, який часто не має таких ресурсів, як великі компанії, але водночас прагне підвищити ефективність через автоматизацію. У таких випадках критичною стає доступність обчислювальних рішень, простота впровадження та пояснюваність результатів. У зв'язку з цим особливої актуальності набувають рекомендаційні системи на основі асоціативних правил, таких як алгоритм Apriori, що вирізняється зрозумілою логікою та невисокі вимоги до обчислювальної потужності [19; 29].

Таким чином, вивчення специфіки функціонування роздрібної торгівлі у поєднанні з новітніми технологіями аналізу даних формує основу для

побудови адаптивних рекомендаційних систем. Ці системи здатні не лише підвищити рівень обслуговування, а й забезпечити переваги в умовах конкуренції за рахунок глибшого розуміння клієнта.

1.2 Проблеми персоналізації в роздрібній торгівлі

Персоналізація в торгівлі, особливо в онлайн-середовищі, стала ключовим інструментом для покращення сервісу, збільшення продажів і покращення взаємодії з клієнтом. Однак, незважаючи на її переваги, впровадження персоналізованих підходів супроводжується рядом складностей і викликів.

Однією з основних проблем є обмеженість або нерелевантність даних про користувача. Для ефективної персоналізації необхідно володіти великою кількістю точних і актуальних даних про поведінку клієнта, його інтереси, історію покупок тощо. Проте нові або неавторизовані користувачі часто не мають достатнього цифрового сліду, що ускладнює формування персоналізованих рекомендацій.

Цю проблему часто називають «проблемою холодного старту», і вона є однією з найактуальніших у побудові рекомендаційних систем. Щоб її подолати, застосовують кілька стратегій: початкове анкетування, використання демографічної інформації, а також залучення контентних характеристик товарів. Наприклад, навіть без історії покупок користувача система може формувати рекомендації на основі переглянутих товарів або часу, проведеного на певній сторінці.

Інша проблема полягає у забезпеченні конфіденційності та безпеки персональних даних. Збирання, зберігання та обробка інформації про користувачів регламентується законодавчими нормами (наприклад, GDPR в Європі) [5], що накладає суттєві обмеження на розробників систем персоналізації. Надмірна або несанкціонована персоналізація також може викликати недовіру з боку споживачів.

Крім того, існує потреба у формуванні прозорих рекомендацій. Користувачі прагнуть розуміти, чому система радить той чи інший товар. Саме тому сучасні системи дедалі частіше впроваджують механізми пояснюваності (explainability), які не лише підвищують довіру, а й допомагають користувачеві краще взаємодіяти з інтерфейсом.

Ще одним викликом є проблема надмірної персоналізації, коли система занадто звужує коло пропозицій, орієнтуючись лише на вузький спектр інтересів користувача. Такий ефект може призвести до зниження відкриття нових товарів (ефект "інформаційної бульбашки") та обмеження асортименту.

Щоб уникнути цього, розробники застосовують баланс між точністю й різноманітністю рекомендацій. Наприклад, додають випадкові або трендові позиції до рекомендованого списку, створюють спеціальні «категорії відкриття» для товарів, які виходять за межі звичних уподобань користувача. Такі підходи дозволяють зберегти елемент новизни та підвищити загальну ефективність системи.

Технічні складнощі також відіграють важливу роль. Це і потреба в обчислювальних ресурсах, і складність побудови ефективних моделей, які можуть адаптуватися до змін у поведінці споживачів у режимі реального часу.

Особливо це стосується систем, що працюють із великими обсягами даних. Обчислення рекомендацій у реальному часі вимагає не лише потужного серверного середовища, а й оптимізованих алгоритмів. З цією метою використовуються методи інкрементального навчання, паралельної обробки даних, а також спеціалізовані бібліотеки (наприклад, TensorFlow, LightFM тощо).

Нарешті, ще однією актуальною проблемою є інтеграція персоналізованих систем у наявну інфраструктуру роздрібного підприємства. Це стосується як технічної сумісності, так і зміни бізнес-процесів, навчання персоналу тощо.

Для успішної інтеграції важливо враховувати не лише технічну сторону, а й організаційні особливості: хто відповідає за оновлення даних, як швидко система реагує на зміну асортименту, які зв'язки існують між відділом IT та маркетингом. Усе це визначає ступінь ефективності персоналізації на практиці.

Таким чином, попри значні можливості, які надає персоналізація, її ефективне впровадження потребує всебічного підходу, який включає технічні, етичні та організаційні аспекти.

1.3 Необхідність використання рекомендаційних систем

У сучасній роздрібній торгівлі, особливо в електронній комерції, кількість доступної продукції постійно зростає, що створює труднощі для споживачів у процесі вибору. Водночас компанії прагнуть забезпечити максимально персоналізовану взаємодію з кожним клієнтом, збільшуючи лояльність та конверсію. У таких умовах рекомендаційні системи набувають особливої ролі, що поєднує інтереси обох сторін — споживача та продавця.

Застосування рекомендаційних систем дозволяє суттєво покращити якість процесу прийняття рішень користувачами, пропонуючи ті товари, які найімовірніше відповідають їхнім інтересам або актуальним потребам. Це зменшує час на пошук і підвищує ймовірність здійснення покупки, що напряму впливає на зростання прибутковості компанії.

Крім того, рекомендаційні системи сприяють оптимізації маркетингових стратегій: вони дозволяють точно сегментувати аудиторію, розробляти індивідуальні пропозиції та знижки, підвищувати ефективність розсилок і рекламних кампаній.

В умовах зростання вартості залучення клієнтів персоналізація комунікацій є вирішальним чинником утримання споживача. Дані про минулі покупки, частоту відвідувань сайту, поведінкові патерни дозволяють

створювати релевантні пропозиції в режимі реального часу. Саме такі механізми демонструють вищу конверсію, ніж традиційна контекстна реклама або масові розсилки.

У технічному плані такі системи забезпечують глибший аналіз клієнтської поведінки, виявляють приховані закономірності у даних і відкривають нові можливості для ухвалення стратегічних рішень. Вони також сприяють покращенню товарної логістики, формуванню оптимального асортименту та ефективнішому управлінню запасами.

Наприклад, аналіз попередніх комбінацій покупок дозволяє прогнозувати попит на товари у зв'язці, що лежить в основі моделей асоціативних моделей на кшталт Apriori. Це дає змогу не лише рекомендувати супутні товари, але й вдосконалити розташування продукції на складах і у фізичних магазинах.

Особливо актуальним є використання рекомендаційних систем у мультиканальному середовищі, де важливо зберігати цілісність клієнтського досвіду незалежно від каналу продажу — онлайн, офлайн або через мобільні додатки. Персоналізовані рекомендації дозволяють забезпечити безшовну інтеграцію між усіма платформами.

У такому контексті зростає роль так званих омніканальних платформ, які поєднують дані з усіх джерел у єдиний профіль користувача. Такий підхід відкриває можливість не лише синхронізувати рекомендації на різних пристроях, але й будувати довгострокову взаємодію з клієнтом, засновану на повному життєвому циклі споживання.

Таким чином, впровадження рекомендаційних систем у сфері роздрібної торгівлі не є виключно інструментом підвищення конкурентоспроможності, а перетворюється на критично важливий елемент ефективної діяльності бізнесу в умовах цифрової трансформації.

1.4 Постановка задачі

В умовах активної цифрової трансформації бізнес-процесів та зростання обсягів доступної інформації, підприємства у сфері роздрібної торгівлі стикаються з необхідністю ефективного застосування інформації про дії користувачів задля підвищення якості обслуговування. Одним із найбільш перспективним напрямом вважається інтеграція систем персоналізованих рекомендацій, які здатні аналізувати історію покупок та вміст поточного кошика, щоби запропонувати найбільш релевантні товари.

Основне завдання даної роботи полягає у побудові інтелектуального механізму, що на основі історичних транзакцій автоматично формує персоналізовані товарні пропозиції для користувача. Така система повинна орієнтуватися на ті товари, які вже були додані до кошика, та використовувати закономірності попередніх покупок інших клієнтів. При цьому система має не лише забезпечувати високу швидкодію, але й бути інтуїтивно зрозумілою для персоналу та справді корисною для покупця, підвищуючи рівень задоволеності клієнтів і сприяючи зростанню продажів.

Для реалізації цього завдання необхідно виконати низку взаємопов'язаних етапів:

1. Створити базу даних, яка акумулюватиме інформацію про покупки користувачів, включаючи склад замовлень, дати, час та ідентифікатори клієнтів, що дозволить здійснювати подальший аналіз і навчання алгоритмів.
2. Реалізувати механізм обліку транзакцій у програмному забезпеченні, забезпечивши повноту й актуальність даних про всі операції.
3. Обрати та адаптувати алгоритм асоціативного аналізу, який дозволить ефективно виявляти часто повторювані поєднання товарів у замовленнях. Наприклад, це може бути класичний алгоритм Apriori або більш оптимізований FP-Growth.

4. Розробити механізм формування рекомендацій, який базуватиметься на отриманих асоціативних правилах і працюватиме в режимі реального часу, адаптуючись до вмісту поточного кошика користувача.
5. Створити інтерфейс для відображення рекомендацій, який буде зручним для сприйняття кінцевим користувачем і легко інтегруватиметься в існуючі канали взаємодії (наприклад, веб-інтерфейс або мобільний додаток).

Отже, виконання заданого завдання дасть змогу не просто підвищити зручність для користувача, а й забезпечити додаткову цінність для бізнесу за рахунок інтелектуальної обробки даних та автоматизації процесу персоналізації у роздрібній торгівлі.

Висновки до розділу 1

У першому розділі було проведено аналіз предметної області, у межах якого охарактеризовано сучасні умови функціонування роздрібної торгівлі та обґрунтовано доцільність застосування персоналізованих рекомендаційних механізмів. Детально розглянуто специфіку взаємодії між продавцем і покупцем у цифровому середовищі, визначено ключові труднощі, з якими стикаються компанії під час впровадження інструментів персоналізації, зокрема проблему обмеженості даних, вимоги до конфіденційності та складність інтеграції в наявні бізнес-процеси.

З'ясовано, що рекомендаційні системи набули статусу необхідного компонента комерційної інфраструктури, особливо за умов інтенсивної конкуренції за увагу споживача. Наголошено на важливості персоналізованого підходу як чинника підвищення лояльності клієнтів та зростання конверсії.

У межах постановки задачі сформульовано основну дослідницьку мету — створення системи, здатної на основі історичних даних формувати

релевантні рекомендації товарів. Було визначено логіку побудови майбутньої програми та окреслено послідовність етапів її реалізації, з акцентом на адаптацію до умов реального торгового середовища з мінімальним втручанням у бізнес-інфраструктуру.

Таким чином, перший розділ заклав теоретичне підґрунтя для подальшого аналізу методів реалізації систем рекомендацій і визначив вектор практичної реалізації.

РОЗДІЛ 2. АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ ТА ВИБІР МЕТОДУ

2.1 Класифікація рекомендаційних систем

Рекомендаційні системи є важливими інструментами, що дозволяють значно підвищити ефективність взаємодії між споживачем і постачальником товарів чи послуг. Вони базуються на аналізі великої кількості інформації про дії користувачів та їхні вподобання. На основі цього аналізу система здатна формувати персоналізовані пропозиції, які максимально відповідають інтересам кожного окремого споживача. Однак існує багато різних підходів до їх побудови, що зумовлює існування численних класифікацій цих систем.

Загалом, рекомендаційні системи можна поділити на три основні типи, залежно від методів, які вони використовують для генерації рекомендацій: фільтрація за змістом, колаборативна фільтрація та гібридні системи. [10; 18; 24]

1. Фільтрація за змістом (Content-based filtering) вважається однією з найпростіших для сприйняття методик. Вона полягає у використанні інформації про самі товари чи послуги для формування рекомендацій. Система аналізує характеристики обраних користувачем товарів (наприклад, їх опис, категорії, характеристики) і шукає інші продукти, що мають схожі ознаки. Такий підхід дозволяє забезпечити точність рекомендацій, однак його головним недоліком є обмеженість в охопленні асортименту, оскільки система не враховує вподобання інших користувачів.

2. Колаборативна фільтрація базується на аналізі спільних уподобань різних користувачів. Вона припускає, що якщо два користувачі мають схожі вподобання щодо деяких товарів, то ймовірно, що їхні інтереси збігаються і для інших продуктів. Такий підхід розподіляється на два основних типи: пористе користування (user-based), де рекомендації будуються на основі схожості користувачів, і пористе товарне (item-based), коли рекомендації

формуються на основі схожості товарів, які були обрані іншими користувачами. Колаборативна фільтрація дозволяє отримати рекомендації без потреби в глибокому аналізі самих товарів, однак вона має проблему "холодного старту", що полягає в необхідності збору достатньої кількості інформації щодо нових користувачів чи товарів. [22; 23; 25]

3. У гібридних системах комбінуються методи фільтрації за змістом і колаборативної фільтрації, намагаючись використати переваги обох підходів. Вони можуть інтегрувати дані про товари, користувачів і їх взаємодію, створюючи більш точні й різноманітні рекомендації. Гібридизація дозволяє знижувати ризик виникнення проблеми "холодного старту" та забезпечує кращу здатність системи реагувати на зміну користувацьких уподобань.[18; 25]

Кожен з перелічених підходів володіє певними плюсами й мінусами, які впливають на їх застосування залежно від конкретних умов і потреб системи. Наприклад, методи фільтрації за змістом забезпечують точність рекомендацій при аналізі добре структурованої інформації, однак вони можуть виявитися неефективними при обробці великих обсягів даних без детальної категоризації. Колаборативна фільтрація, з іншого боку, здатна ефективно генерувати рекомендації без необхідності детального аналізу властивостей товарів, але має обмеження в ситуаціях, коли користувачі або продукти нові. Гібридні методи, комбінуючи переваги обох підходів, дозволяють досягти високої точності й універсальності системи, хоча й потребують значних обчислювальних ресурсів.

Отже, класифікація рекомендаційних систем за використовуваними методами дозволяє окреслити основні напрями розвитку цієї технології. Кожен підхід має свою роль у сучасній роздрібній торгівлі і торгових системах, і правильний вибір методу залежить від специфіки продуктів, поведінки споживачів та технічних можливостей системи.

2.2 Колаборативна фільтрація

У контексті побудови сучасних рекомендаційних систем метод колаборативної фільтрації залишається одним із найпоширеніших завдяки своїй здатності формувати персоналізовані пропозиції без потреби в аналізі змісту самих об’єктів. Основна ідея цього підходу ґрунтується на припущенні, що користувачі з подібними вподобаннями в минулому, ймовірно, проявлять інтерес до однакових об'єктів у майбутньому.

Умовно колаборативну фільтрацію можна поділити на два напрямки: орієнтовану на користувачів (user-based) і орієнтовану на об’єкти (item-based). У першому випадку система формує рекомендації, аналізуючи поведінкові патерни користувачів, які виявляють схожість у виборі товарів або оцінках. Такий підхід дозволяє, наприклад, визначити групу користувачів зі схожими історіями покупок і запропонувати одному з них товар, який був обраний іншими членами цієї групи, але ще не з’явився у його кошику.

Таблиця 2.1 – Порівняння основних підходів колаборативної фільтрації

Критерій	User-based фільтрація	Item-based фільтрація
Орієнтація	На подібність між користувачами	На подібність між об'єктами (товарами)
Джерело інформації	Поведінкові дані користувачів	Історія одночасних покупок товарів
Переваги	Персоналізовані рекомендації	Стабільність при великій кількості користувачів
Недоліки	Погано масштабується, чутливість до шуму	Менш гнучка до нових вподобань
Чутливість до «холодного старту»	Висока (нові користувачі)	Висока (нові товари)
Придатність для великих даних	Менш ефективна	Краще підходить для великих систем

Другий підхід базується на виявленні зв’язків між самими товарами. Якщо, згідно з історичними даними, певні товари часто купуються разом, система може автоматично запропонувати ці пари новим користувачам. Це дозволяє значною мірою зменшити залежність від кількості доступних даних

про конкретного покупця, що особливо важливо у випадку нових користувачів або при фрагментованій історії транзакцій.

Однак, попри низку переваг, колаборативна фільтрація має і відчутні обмеження. Одним з найпоширеніших викликів є так звана проблема "холодного старту", що виникає у разі появи нових користувачів або товарів, для яких ще не накопичено достатньо даних. Окрім цього, зростання обсягу інформації призводить до суттєвих обчислювальних витрат, що може ускладнити масштабування системи без оптимізації алгоритмів.

Попри зазначені недоліки, колаборативна фільтрація залишається ефективним підходом у тих випадках, коли система має доступ до репрезентативного масиву транзакційних даних. За умови достатнього обсягу й структурованості таких даних, алгоритми цього класу здатні формувати рекомендації з високим рівнем релевантності, підвищуючи якість користувацького досвіду й загальну ефективність торгівлі.

2.3 Контентна фільтрація

Контентно-орієнтований підхід до побудови рекомендаційних систем ґрунтується на припущенні, що користувачі, які раніше позитивно оцінювали певні об’єкти, з імовірністю оберуть інші об’єкти зі схожими характеристиками. На відміну від колаборативної фільтрації, яка спирається на поведінкові патерни інших користувачів, контентна фільтрація використовує описові властивості самих об’єктів — зокрема, категорії, теги, ключові слова, текстові описи чи інші релевантні атрибути.

У своїй основі контентна фільтрація використовує векторні представлення об’єктів (наприклад, товарів) і профілю користувача, які будуються шляхом аналізу попередньо вподобаних або придбаних товарів. Кожен об'єкт репрезентується у вигляді багатовимірного вектора, значенням якого відповідають наявність або інтенсивність певних характеристик.

Профіль користувача, у свою чергу, агрегує властивості об’єктів, які раніше були обрані або оцінені ним позитивно. Подальша генерація рекомендацій виконується шляхом обчислення ступеня схожості між новими об'єктами й сформованим профілем.

Перевагою такого підходу є його здатність ефективно працювати у ситуаціях, коли користувацьких даних ще недостатньо, тобто у випадку "холодного старту" для нових користувачів. Крім того, контентна фільтрація дозволяє уникнути проблеми "ефекту популярності", властивої колаборативним методам, і формувати більш індивідуалізовані рекомендації, орієнтуючись на унікальні інтереси конкретної особи.[27]

Водночас, дана методика має й обмеження. Зокрема, вона не здатна враховувати латентні переваги користувача, які не можуть бути виведені з аналізу об’єктивних характеристик товарів. До того ж, у системах, де атрибутивні описи є обмеженими або неструктурованими, виникає проблема побудови якісних векторів ознак. Не менш критичним є також ризик надмірної вузької спеціалізації, коли система починає рекомендувати лише ті об'єкти, які надто схожі на вже вибрані, ігноруючи потенційно цікаві, але відмінні варіанти — так званий "ефект змістовного резонансу".

Попри зазначені труднощі, контентна фільтрація є цінним інструментом у загальному арсеналі методів персоналізації, особливо у випадках, коли наявні структуровані описи товарів та обмежена історія взаємодії користувача з системою. В умовах правильної реалізації цей підхід здатен забезпечити високий рівень релевантності рекомендацій і гнучкість налаштування під специфіку предметної області.

2.4 Гібридні підходи

У контексті зростаючої складності потреб користувачів і стрімкого зростання обсягів доступної інформації, окреме застосування колаборативної

або контентної фільтрації часто виявляється недостатнім для забезпечення належного рівня точності та релевантності рекомендацій. За таких обставин гібридні підходи розглядаються як перспективне рішення, що поєднує переваги кожного з методів, врівноважуючи їхні притаманні обмеження.

Сутність гібридних систем рекомендацій полягає у синтезі декількох незалежних стратегій фільтрації — найчастіше колаборативної та контентної — шляхом інтеграції їхніх результатів або алгоритмічного об’єднання на певному етапі обробки даних. Така інтеграція може реалізовуватися на кількох рівнях: починаючи з поєднання кінцевих оцінок у вигляді зваженої суми, до глибокої взаємної адаптації внутрішніх структур моделей.

Існує кілька типових моделей гібридизації, серед них можна назвати такі основні:

1. Послідовна гібридизація — один метод використовується як фільтр, який формує попередній список кандидатів, тоді як другий — уточнює ранжування.
2. Паралельна гібридизація — результати декількох методів об'єднуються після незалежного виконання, зазвичай через агрегацію оцінок.
3. Інтегрована гібридизація — характеристики та моделі різних методів поєднуються в межах одного алгоритмічного ядра, що передбачає складніші структури, зокрема на базі нейронних мереж або факторизації матриць.[20; 21]

Однією з найсуттєвіших переваг гібридних систем є зменшення ризику прояву "холодного старту", оскільки недоліки одного підходу можуть бути компенсовані сильними сторонами іншого. Наприклад, у випадку обмеженого обсягу історичних даних колаборативна компонента може бути доповнена контентним аналізом, що дозволяє зберігати якість рекомендацій навіть для нових користувачів або товарів.

Натомість, гібридні системи характеризуються підвищеною обчислювальною складністю та вимогливістю до структурованості даних.

Проєктування таких рішень потребує більшої кількості експериментів, тонкого налаштування вагових коефіцієнтів або навчання багатошарових моделей, що вимагає залучення додаткових ресурсів — як технічних, так і аналітичних.

У загальному випадку, гібридні методи демонструють високу гнучкість і адаптивність, що робить їх доцільними в умовах великомасштабних систем електронної комерції. Саме завдяки своїй універсальності вони набули широкого розповсюдження у провідних рекомендаційних платформах сучасності, забезпечуючи індивідуалізований підхід до формування користувацького досвіду.

2.4.1 Аналіз методів інтелектуального аналізу даних, застосовуваних у рекомендаційних системах

Перш ніж обґрунтувати вибір конкретного методу, варто розглянути основні підходи, які застосовуються в інтелектуальному аналізі даних (ІАД) для побудови рекомендаційних систем. Існує низка напрямів, що використовуються для виявлення закономірностей у великих обсягах інформації, кожен із яких має свої особливості, переваги та обмеження.

Методом, який часто використовується на практиці є кластеризація. Її суть полягає у групуванні об’єктів (користувачів або товарів) за подібністю. У контексті рекомендацій це дозволяє формувати спільні профілі для клієнтів з близькими уподобаннями. До популярних алгоритмів належать k-means, DBSCAN, ієрархічна кластеризація. Кластеризація є зручною, коли необхідно сегментувати аудиторію, однак вона не завжди забезпечує точні індивідуальні рекомендації.

Іншим напрямом є класифікація, яка передбачає навчання моделі на основі маркованих даних для передбачення категорій або класів. Найчастіше використовується для прогнозування реакції користувача на новий товар.

Серед популярних алгоритмів — дерева рішень, наївний байєсівський класифікатор, логістична регресія.

Окрему нішу займає асоціативний аналіз, метою якого є виявлення частих поєднань елементів у даних. На противагу кластеризації та класифікації, асоціативні правила не потребують попереднього маркування чи профілювання користувачів, що робить їх особливо привабливими для роботи з транзакційними базами. Цей підхід дозволяє з'ясувати, які товари зазвичай купують разом, і на цій основі формувати персоналізовані рекомендації. До найвідоміших алгоритмів у цьому напрямку належать Apriori, FP-Growth та Eclat.

Порівняльний аналіз показав, що метод асоціативних правил, а саме алгоритм Apriori, найкраще відповідає умовам реалізації рекомендаційної системи для локального магазину: він не потребує глибокого навчання, добре працює з історією замовлень і забезпечує прозорість логіки рекомендацій. Це стало вирішальним фактором при виборі алгоритмічного ядра для даної роботи.

2.5 Асоціативні правила та алгоритм Apriori

У системах, що мають справу з великими обсягами даних про поведінку користувачів, особливо в роздрібній торгівлі, важливе місце посідає виявлення повторюваних шаблонів у виборі товарів. Один із дієвих підходів до цього — застосування асоціативних правил, які дозволяють виявити приховані взаємозв'язки між окремими позиціями у кошиках покупців.[19; 29] Іншими словами, мова йде про пошук залежностей на кшталт: якщо обрано певний товар чи групу товарів, то з високою ймовірністю буде обрано й інший — той, що часто траплявся в подібних комбінаціях у минулих транзакціях.

Для автоматизованого виявлення таких закономірностей найбільш поширеним є алгоритм Apriori, що запропонований Агравалом і Шрікантом у

1994 році. Його популярність пояснюється поєднанням простої реалізації, прозорої логіки роботи та добрих результатів у системах з помірним обсягом транзакцій.

Основна ідея алгоритму полягає у використанні так званої властивості анти-монотонності: якщо певний набір товарів не є частим, то жодна його надмножина теж не може бути частою. Завдяки цій властивості можна значно скоротити кількість перевірок, необхідних для знаходження частих наборів, що дозволяє підвищити ефективність обчислень.

З технічної точки зору, Apriori працює за ітеративною схемою, у якій на кожному кроці генеруються кандидатні набори (k-множини) з попередніх частих (k–1)-множин, після чого виконується їх перевірка на відповідність пороговому значенню support — частоти зустрічання у транзакціях. Процес триває доти, доки не залишиться жодного нового частого набору. Цей підхід дозволяє згенерувати повний перелік частих комбінацій товарів.

Після виявлення таких наборів формується набір асоціативних правил

- Support — частка транзакцій, що містять одночасно і A і B

$$supp(A \to B) = \frac{\sigma(A \cup B)}{N}$$

- Confidence — умовна ймовірність появи B, якщо є A:

$$conf(A \to B) = \frac{\sigma(A \cup B)}{\sigma(A)}$$

- Lift — співвідношення фактичної ймовірності спільної появи A і B до очікуваної при їх незалежності:

$$lift(A \to B) = \frac{conf(A \to B)}{supp(B)}$$

Оскільки зростання кількості унікальних товарів призводить до експоненційного зростання кількості можливих комбінацій, алгоритм Apriori може бути обчислювально затратним, особливо при низьких порогах підтримки. Цей недолік частково усувається за рахунок попереднього

відсікання неперспективних комбінацій, однак у випадках із мільйонами транзакцій його використання стає обмеженим. У таких ситуаціях частіше застосовують альтернативні методи, як-от FP-Growth (Frequent Pattern Growth), який уникає явної генерації кандидатів за рахунок побудови стислого дерева частих шаблонів.

Існує кілька підходів до реалізації Apriori:

- Класичний — із послідовною генерацією k-множин через об'єднання (join) і перевірку підмножин.
- Hash-based — зі збереженням кандидатів у геш-таблицях для швидшого доступу.
- Transaction reduction — коли транзакції, які не можуть містити жодного кандидата, відсікаються на ранніх етапах.
- Partitioning — розбивання всієї бази на частини, знаходження частих наборів у кожній з них окремо, а потім — об'єднання результатів.

У Більшості сучасних мов програмування вже існують бібліотеки для роботи з алгоритмом Apriori (наприклад, mlxtend у Python, arules у R), проте у цій роботі зроблено акцент на власну реалізацію, адаптовану до специфіки невеликої локальної системи з обмеженим обсягом даних і вимогами до швидкодії в реальному часі.

На практиці алгоритм Apriori виявляє високу ефективність при використанні в рекомендаційних системах типу «разом купують», особливо коли ціль — не стільки побудувати складні прогнози, скільки надати користувачу прості, логічно обґрунтовані рекомендації. У таких випадках прозорість логіки формування правила є перевагою, оскільки дозволяє бізнесу пояснити, чому запропоновано саме ці товари, що підвищує довіру до системи.

У рамках цієї роботи реалізація Apriori була адаптована до конкретного завдання — формування рекомендацій на основі частих комбінацій у історії замовлень, причому кожне правило будується динамічно на основі поточного вмісту кошика. Такий підхід, з одного боку, він залишається простим і

гнучким, а з іншого — ефективно забезпечує достатню швидкість роботи та адаптивність до нових транзакцій.

2.6 Обґрунтування вибору методу для реалізації

У процесі проектування ефективної рекомендаційної системи постає ключове питання вибору алгоритмічного підходу, що дозволив би не лише формувати персоналізовані пропозиції, але й робити це з урахуванням обмеженого обсягу транзакційних даних, притаманного малим і середнім суб'єктам роздрібної торгівлі. У цьому контексті постає необхідність обрати метод, який з одного боку, він відзначається концептуальною прозорістю, а з іншого — має здатність виявляти приховані закономірності між елементами даних без потреби у глибокому навчанні чи попередньому маркуванні вибірки.

Серед широкого спектра існуючих підходів — від латентних факторних моделей до гібридних нейромережевих архітектур — метод асоціативних правил виглядає найбільш релевантним щодо поставленої задачі. Його перевага полягає в тому, що логіка формування рекомендацій базується на частотності співпадіння товарів у попередніх замовленнях, що є інтуїтивно зрозумілим як для розробника, так і для кінцевого користувача. Важливо, що цей метод не лише виявляє парні або множинні асоціації між об'єктами (товарами), але й дозволяє інтерпретувати результати в категоріях практичної доцільності.

Особливу увагу при реалізації було зосереджено на адаптації базових принципів алгоритму Apriori, але без його класичної обчислювальної складності. Замість того, щоб здійснювати повний перебір усіх можливих комбінацій було запропоновано використовувати евристичну перевірку повного входження обраного набору товарів у історичні транзакції інших користувачів. Подібна стратегія дозволяє зберегти аналітичну глибину при

зниженні вимог до обчислювальних ресурсів, що є критичним фактором у середовищі настільного застосунку з локальною базою даних.

Покроковий опис адаптованого алгоритму Apriori

Крок 1. Формування поточного кошика

Користувач додає до кошика товари T = {t_1, t_2, …, tn}.

Крок 2. Пошук транзакцій

Із бази історичних замовлень D вибираються всі транзакції T_i, які містять усі товари з кошика.

$$T_i \in D, \qquad \text{де } T_{user} \subseteq T_i$$

Крок 3. Аналіз доповнень

У знайдених транзакціях підраховуються всі інші товари, що найчастіше зустрічаються разом із поточним набором.

Крок 4. Обчислення підтримки

Для кожного товару x, що не входить до кошика, обчислюється підтримка як відношення кількості входжень до кількості релевантних транзакцій.

$$supp(x) = \frac{\text{кількість входжень x у релевантних транзакціях}}{\text{загальна кількість релевантних транзакцій}}$$

Крок 5. Фільтрація (опційно)

Можна ввести поріг підтримки або просто впорядкувати товари за популярністю.

Крок 6. Формування рекомендацій

Видаються товари з найвищою підтримкою як рекомендації до поточного кошика.

R = {x1,x2,...,xk}, де supp(x_i) > supp(x_j) при i < j

Такий підхід не потребує побудови всіх можливих комбінацій товарів, як у класичному Apriori, і дозволяє швидко формувати релевантні рекомендації, спираючись на історичну поведінку користувачів.

Таким чином, зроблений вибір не є компромісом між простотою та точністю, а радше — результатом рефлексивного зіставлення функціональних вимог, обмежень платформи реалізації (Windows Forms, SQL LocalDB) та прагнення до максимальної інтерпретованості результатів. Такий підхід також забезпечує відкритість до подальшої модифікації або масштабування у бік складніших алгоритмічних рішень, зокрема на базі моделі Item-based Collaborative Filtering або кластеризації на основі поведінкових профілів, якщо у майбутньому з'являться відповідні ресурси та обсяги даних.

Висновки до розділу 2

У другому розділі розглядаються сучасні методи реалізації побудови рекомендаційних систем у контексті інтелектуального аналізу даних. Порівняно ключові напрями — кластеризацію, класифікацію, фільтрацію та асоціативні правила — з погляду їх ефективності, гнучкості, вимог до даних і можливостей адаптації до локального середовища. Особливу увагу зосереджено на методах асоціативного аналізу як найбільш релевантних до умов задачі, що не потребують глибокого навчання моделей або використання особистісних характеристик користувачів.

На цьому тлі алгоритм Apriori виявився оптимальним як з позиції технічної реалізації, так і з точки зору інтерпретованості результатів. Детально розглянуто механізм його роботи, ключові метрики (support, confidence, lift), а також надано огляд основних підходів до його реалізації — від класичної до оптимізованих варіантів.

У межах підрозділу 2.6 обґрунтовано вибір саме цього алгоритму для побудови рекомендаційної системи враховуючи обмеження обчислювальних

ресурсів, характер даних і потребу у швидкодії. Запропоновано модифікацію алгоритму з орієнтацією на евристичне порівняння вмісту кошика з історичними транзакціями, що дозволяє зберегти логіку Apriori, але адаптувати її до конкретного прикладного контексту.

Таким чином, розділ 2 заклав методологічну основу для реалізації системи персоналізованих рекомендацій, поєднуючи аналітичну глибину з практичною орієнтованістю на потреби малого роздрібного бізнесу.

РОЗДІЛ 3. РОЗРОБКА ТА РЕАЛІЗАЦІЯ РЕКОМЕНДАЦІЙНОЇ СИСТЕМИ

3.1 Архітектура програмного забезпечення

Програмне рішення побудоване за принципом локальної автономності, де ключова ставка зроблена на незалежність від зовнішніх сервісів. У його основі лежить архітектура, що передбачає безпосередню взаємодію між користувачем та локально розміщеною базою даних.[3; 13] Такий підхід дозволяє уникнути залежності від стабільності інтернет-з'єднання, що критично важливо для невеликих підприємств, які працюють у сфері роздрібної торгівлі.

Застосунок реалізовано на основі платформи Windows Forms із використанням C#. У ролі сховища даних виступає вбудована база .mdf, яка підключається до проєкту безпосередньо, минаючи потребу в налаштуванні окремого SQL-сервера. Це рішення спрощує розгортання системи й робить її придатною для використання навіть у локальних середовищах без додаткової інфраструктури.

Функціональна структура програмного забезпечення складається з кількох самостійних, але взаємопов'язаних частин. Користувацький інтерфейс, винесений у дві основні форми — вхідну (автентифікаційну) та основну (торгову) — забезпечує інтуїтивну взаємодію з системою. Усередині головної форми відображається вітрина товарів, модуль кошика та блок рекомендацій, що генеруються динамічно.

За логіку аналізу відповідає окремий функціональний блок, який проводить обробку історичних даних та на їх основі пропонує товари, що потенційно можуть зацікавити користувача. Усі дії зберігаються у внутрішній структурі бази даних, де таблиці замовлень, транзакцій і користувачів взаємодіють у режимі реального часу.

Рисунок 3.1 – Схема взаємодії компонентів програмної системи

Такий підхід дозволив створити систему, що поєднує простоту, автономність і інтелектуальну адаптивність до поведінки користувача, при цьому залишаючись технічно легкою у впровадженні.

3.2 Вибір середовища розробки та інструментів

Під час розробки програмного забезпечення рекомендаційної системи ключовим етапом стало визначення оптимального набору інструментів, які б забезпечили ефективну реалізацію як клієнтської логіки, так і взаємодії з базою даних. Враховуючи специфіку задачі — створення настільного додатку із зручним графічним інтерфейсом та інтеграцією з локальною SQL-базою — вибір було зосереджено на екосистемі Microsoft .NET.

Основним середовищем розробки обрано Microsoft Visual Studio, яке надає широкі можливості для побудови Windows Forms-додатків.[2; 6; 12] Це

середовище поєднує інтегровані інструменти для візуального проектування інтерфейсів, зручне налагодження, управління проєктами, а також підтримку популярних мов програмування, зокрема C#.

Для реалізації основної логіки програми використано мову C#, що є високорівневою, строго типізованою мовою із широким спектром бібліотек. Завдяки своїй об'єктно-орієнтованій природі, C# дозволяє чітко структурувати програмний код, ізолювати бізнес-логіку та сприяти підтримуваності системи.

Для забезпечення якості системи керування базами даних було обрано Microsoft SQL Server LocalDB, який забезпечує легку інтеграцію з .NET-додатками, можливість локального зберігання даних і просте розгортання у середовищі розробника. Для роботи з БД використовувався простий механізм доступу через SqlConnection, що забезпечує пряме виконання SQL-запитів з боку програми.

Для візуального представлення даних і побудови інтерактивного інтерфейсу активно використовувались елементи Windows Forms: FlowLayoutPanel, Panel, Label, PictureBox, Button тощо. Додатково, з метою покращення користувацького досвіду, були застосовані візуальні ефекти: зміна кольору при наведенні, динамічне створення карток товарів, кастомізація кнопок.

Таким чином, обрані інструменти повністю відповідали вимогам до функціональності системи, гнучкості реалізації і забезпечили баланс між продуктивністю та зручністю розробки.

3.3 Структура бази даних транзакцій

Формування структурованої і водночас гнучкої бази даних є передумовою стабільного функціонування будь-якої інформаційної системи, що обробляє поведінкові дії користувачів у контексті електронної комерції. У рамках реалізації програмного забезпечення, що включає механізм

рекомендацій, особлива увага була приділена формуванню сховища для запису транзакцій та супутньої інформації про товари й облікові записи.

База даних створена на основі реляційної моделі та складається з трьох основних таблиць: Users, Products і Orders.

OrderId	UserId	Products	CreatedAt
1	1	Борщ українсь...	02.05.2025 9:47:...
2	3	Борщ українсь...	02.05.2025 10:0...
3	2	Хот-дог класич...	02.05.2025 10:5...
4	4	Піцца Маргар...	02.05.2025 10:5...
5	2	Мафін шокола...	02.05.2025 10:5...
6	3	Сік апельсино...	02.05.2025 10:5...

Id	Name	Price	ImagePath
1	Кава латте	49,99	Images\latte.jpg
2	Хот-дог класич...	35,00	Images\hotdog....
3	Борщ українсь...	55,00	Images\borsch.j...
4	Круасан з шок...	25,50	Images\croissa...
5	Капучино	47,00	Images\cappuc...
6	Піцца Маргар...	89,00	Images\pizza.jpg
7	Сендвіч з курк...	52,00	Images\chicken...

Id	Username	Password
1	Влад	аРп1256
2	maxim	Nxr123455
3	Максим	Макс1234
4	Ева	Мпра567
5	Іра	М123123

Рисунок 3.2-3.4 – Таблиці Users, Products і Orders

Кожна з них виконує окрему функціональну роль і пов’язана з іншими логічними залежностями. Так, таблиця Users містить ідентифікаційні дані користувача: унікальний ідентифікатор (Id), логін та пароль. Таблиця Products — центральне джерело інформації для візуалізації карток товарів — включає поля для збереження назви товару, ціни та шляху до зображення.

Проте найбільш цікавою у контексті побудови рекомендаційної логіки є таблиця Orders. Саме вона фіксує факти взаємодії користувача з системою шляхом оформлення замовлень. Структура цієї таблиці, спочатку може здатися простою: вона включає поле UserId, яке вказує на замовника, та поле Products, у якому зберігається перелік придбаних товарів у текстовому форматі. Однак саме цей формат, що передбачає фіксацію назв товарів разом із кількістю (наприклад, «Круасан×2, Лимонад×1»), дає змогу реалізувати гнучкий підхід до аналізу транзакцій без складних джойн-запитів.

З технічного погляду, таке рішення дозволяє зберігати комбінації товарів у вигляді цілісного рядка, що спрощує подальший синтаксичний розбір за допомогою засобів C# на стороні клієнта. Незважаючи на те, що такий підхід дещо ускладнює реалізацію глибоких SQL-аналітичних запитів, він цілком виправданий для систем, у яких основна логіка обробки виконується на рівні бізнес-логіки програми.

З метою розширення функціоналу у майбутньому можливе впровадження додаткових таблиць, таких як Transactions, для збереження кожної дії користувача (наприклад, додавання до кошика), що стане підґрунтям для детальнішої персоналізації та точного трекінгу дій.

Варто зазначити, що реалізована структура БД демонструє не лише відповідність поточним вимогам, але й готовність до масштабування. Наприклад, при потребі впровадження повноцінного алгоритму Apriori із зважуванням частоти — можна здійснити реконструкцію схеми в бік більшої нормалізації.

3.4 Реалізація алгоритму Apriori

У процесі створення системи персоналізованих рекомендацій виникла потреба не лише у привабливому інтерфейсі, а й у внутрішньому логічному механізмі, здатному обробляти історичні дані про замовлення користувачів з метою виявлення повторюваних товарних поєднань. Головною вимогою до цього механізму було те, щоб він міг працювати швидко, адаптуватися до нових даних і водночас залишатися прозорим для розробника та зрозумілим з позиції логіки поведінки.

Після аналізу можливих підходів, доцільним виявилось використання методу асоціативного аналізу, зокрема алгоритму Apriori. Його перевага полягає в здатності знаходити стійкі закономірності у поєднаннях товарів без залучення персональних характеристик користувачів або навчання на маркованих вибірках. Втім, класичне застосування Apriori передбачає перебір усіх можливих комбінацій товарів і поступову фільтрацію за частотою, що у реальних умовах виявилось надто ресурсоємним і непридатним для інтерактивної системи.

Зважаючи на це, було вирішено адаптувати ідею алгоритму до більш практичної моделі. Основна логіка збереглась: аналіз транзакцій для виявлення часто повторюваних супутніх товарів. Проте реалізація ґрунтується на перевірці повного входження товарів із поточного кошика користувача до минулих замовлень, що зберігаються у базі. Якщо таке входження виявлено, система аналізує інші товари, присутні в цій транзакції, та підраховує частоту їх появи в релевантних замовленнях. На основі цих підрахунків обчислюється підтримка для кожного товару — як частка релевантних транзакцій, у яких він зустрічається, що забезпечує об’єктивну оцінку сили асоціації з поточним кошиком.

Якщо таке входження виявлено, система аналізує інші товари, присутні в цій транзакції, та підраховує частоту їх появи в релевантних замовленнях. На основі цих підрахунків обчислюється підтримка для кожного товару — як частка релевантних транзакцій, у яких він зустрічається, що забезпечує

об’єктивну оцінку сили асоціації з поточним кошиком. Щоб уникнути врахування випадкових або слабких зв’язків, до рекомендацій включаються лише ті товари, чия підтримка перевищує заданий поріг (Supp_min), що у реалізації встановлений на рівні 20%. З нього система обирає три найпоширеніші й подає їх у вигляді карток рекомендацій.

Рисунок 3.5 – Алгоритм формування рекомендацій

На практиці реалізацію цього підходу зосереджено у методі UpdateRecommendations(). Його логіка передбачає кілька основних кроків:

- отримання товарів із поточного кошика;
- вибірка історичних замовлень з бази даних (крім замовлень поточного користувача);
- обробка кожного запису для виявлення входження всіх товарів із кошика;
- накопичення частоти появи додаткових товарів;
- сортування результатів і виведення трьох найбільш релевантних.

Кожне замовлення в базі зберігається у вигляді текстового рядка, де товари відокремлено комами, а кількість кожного позначено через символ множення (наприклад: Кава×2). Це дозволило зберегти просту структуру збереження без побудови складної нормалізованої схеми, що було важливо для локальної системи.

Цей підхід володіє низкою переваг. Насамперед, рекомендації формуються одразу — система не потребує попереднього навчання або ручного оновлення моделі. По-друге, оскільки вся логіка працює на рівні простих множин та частотного підрахунку, результат можна пояснити — це важливо як для користувача, так і для подальшого вдосконалення системи. По-третє, адаптація під локальне середовище дозволяє використовувати її без підключення до інтернету, що важливо для невеликих закладів, які не мають постійної хмарної інфраструктури.

Отже, реалізований алгоритм зберігає концептуальну основу методу Apriori, але водночас є легким, швидким та ефективним рішенням для практичної задачі рекомендацій у середовищі з обмеженим обсягом даних. Такий варіант поєднує аналітичну логіку з реалістичним підходом до розробки програмного забезпечення, демонструючи, як на основі теоретичних алгоритмів можна побудувати прикладний інструмент з практичним ефектом.

3.5 Формування рекомендацій для користувача

Одним із ключових етапів у функціонуванні рекомендаційної системи є процес формування персоналізованих пропозицій, що ґрунтуються на наявних даних про попередні покупки інших користувачів. У межах даної реалізації було побудовано механізм, який дозволяє генерувати релевантні рекомендації на основі транзакційної історії, використовуючи наближену логіку алгоритму асоціативних правил (Apriori).

Рекомендації формуються одразу після додавання хоча б одного товару до кошика. Алгоритм, що реалізований у програмі, аналізує комбінації товарів, збережені у базі даних у вигляді рядків замовлень. Кожне замовлення представлено як перелік товарів з вказаною кількістю, що ускладнює обробку, але надає повніше уявлення про поведінку споживача. Для побудови рекомендацій програма ідентифікує ті замовлення інших користувачів, що повністю включають усі товари, наявні в поточному кошику. Із цих релевантних замовлень витягуються товари, відсутні в кошику, та ведеться підрахунок частоти їхньої появи.

Після завершення аналізу частотності система виводить користувачеві до трьох найбільш часто співпадаючих товарів у вигляді окремих карток. У випадках, коли жодне з історичних замовлень не містить повного збігу з поточним вмістом кошика, система автоматично переходить до режиму fallback і пропонує найбільш популярні товари, що найчастіше траплялися в усіх замовленнях інших користувачів.

Важливим аспектом реалізації є динамічне оновлення рекомендацій у режимі реального часу: кожна зміна вмісту кошика (додавання або видалення товару) миттєво ініціює повторне виконання алгоритму. Такий підхід забезпечує адаптивність системи до дій користувача і дозволяє формувати більш релевантні й контекстуальні пропозиції.

Додатково, система підтримує механізм візуалізації асоціацій між товарами: за допомогою окремого інтерфейсного елементу можна переглянути, які саме товари найчастіше зустрічаються у комбінаціях з

поточним вмістом кошика. Це підвищує прозорість роботи алгоритму та сприяє кращому розумінню логіки рекомендаційного механізму з боку користувача.

Таким чином, реалізований механізм формування рекомендацій поєднує простоту, адаптивність до поточних дій користувача та базується на практичних засадах інтелектуального аналізу даних, що забезпечує високу ефективність у контексті невеликих транзакційних баз.

3.6 Інтерфейс користувача

Проєктування інтерфейсу користувача є невід'ємною складовою розробки програмного забезпечення, орієнтованого на кінцевого споживача. У контексті створення рекомендаційної системи для роздрібної торгівлі основною метою стало не лише забезпечення коректного виконання функцій, а й досягнення високого рівня зручності, доступності та візуальної привабливості. Успішна реалізація інтерфейсу значною мірою визначає ефективність взаємодії між користувачем та алгоритмічним ядром системи.

Після авторизації або реєстрації, користувач потрапляє на головну форму додатку, де здійснюється взаємодія з функціональністю системи. Інтерфейс має чітке логічне структурування: у верхній частині форми розташовано елементи навігації або виведення інформації про активного користувача, тоді як центральна область присвячена відображенню переліку товарів. Список формується динамічно на основі даних з бази, що забезпечує актуальність відображуваного асортименту.

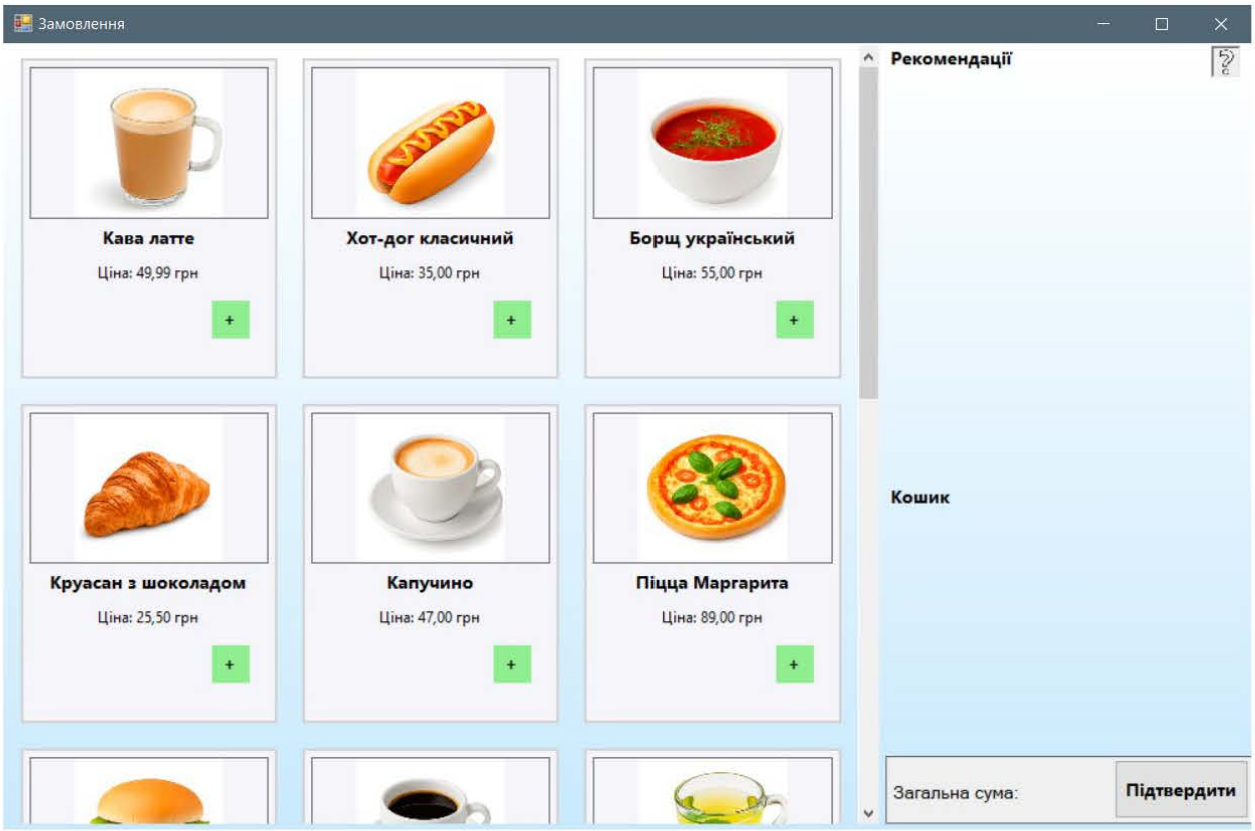

Рисунок 3.5. Головна форма

Товари представлені у вигляді карток. Кожна картка містить зображення товару, його назву, вартість, а також кнопку для швидкого додавання до кошика. При наведенні на елементи картки змінюється фон, що сигналізує про можливість взаємодії. Кнопка «+» дозволяє миттєво додати товар, не переходячи на інші екрани чи підтвердження.

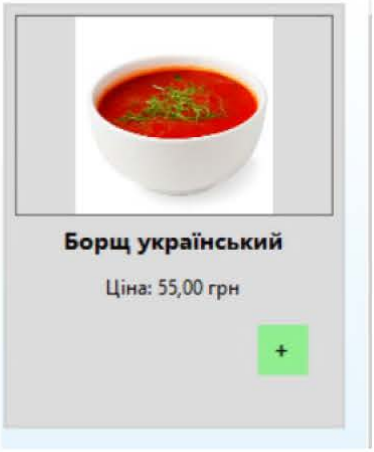

Рисунок 3.6 – Приклад карток товарів у списку

Кошик реалізовано як окрему панель, що розташована праворуч. Він відображає список доданих товарів із вказаною кількістю, індивідуальною сумою та загальною сумою замовлення. Кожен товар у кошику має кнопку для зменшення кількості або повного видалення. У реальному часі система оновлює інформацію про загальну вартість покупки, що є ключовим з погляду прозорості транзакції.

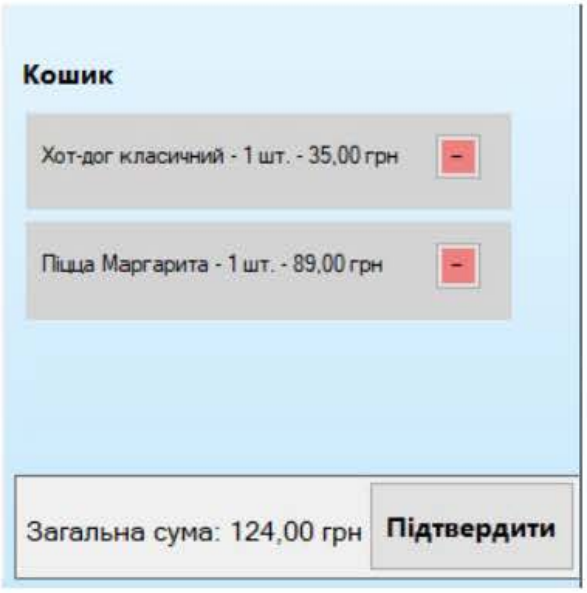

Рисунок 3.7 – Кошик із доданими товарами

Окремим елементом виступає модуль рекомендацій. Він реалізований у нижній частині форми у вигляді горизонтальної панелі з картками рекомендованих товарів. Якщо система знаходить релевантні рекомендації на основі історії схожих покупок інших користувачів, вони відображаються у відповідному блоці. У разі відсутності релевантних збігів, система підбирає популярні товари. Механізм реактивної побудови рекомендацій дозволяє змінювати пропозиції в реальному часі під час додавання чи видалення товарів з кошика.

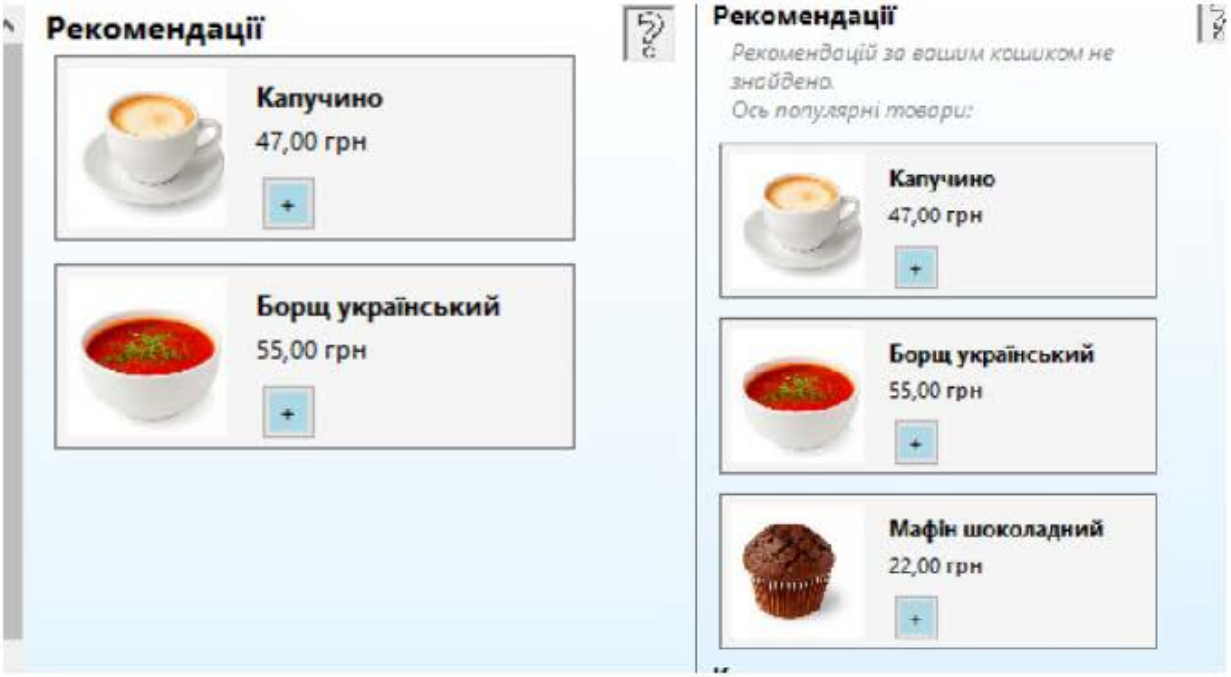

Рисунок 3.8 - 3.9 –Блок з персоналізованими рекомендаціями.

Також реалізовано можливість перегляду частотних зв'язків між товарами, що дозволяє користувачу краще зрозуміти логіку рекомендацій. Ця функція активується натисканням на окрему іконку, після чого відкривається таблиця, яка містить товари, що найчастіше зустрічались разом з тими, що вже в кошику. Це сприяє підвищенню довіри до системи рекомендацій і створює додаткову цінність для користувача.

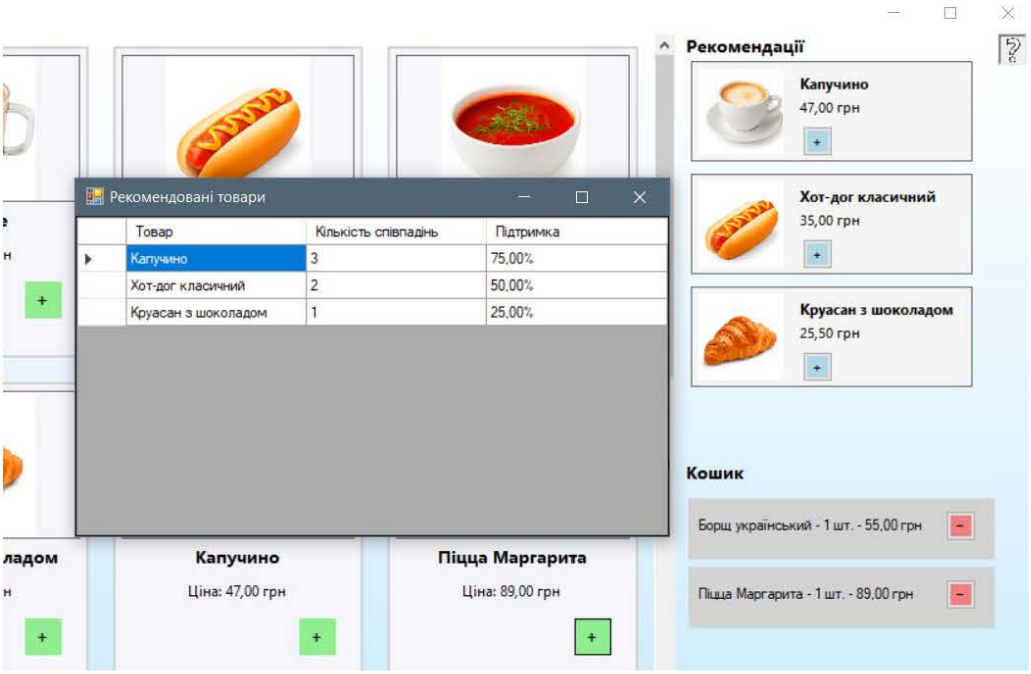

Рисунок 3.10 – Таблиця співпадінь товарів, що зустрічаються разом у замовленнях.

Усі компоненти інтерфейсу оформлені з урахуванням сучасних вимог до UI/UX. Застосовано градієнтне тло, світлі кольори, плавні тіні та зрозумілі шрифти. Реакція на події миші (наведення, натискання) забезпечує динамічну та приємну взаємодію.

Таким чином, інтерфейс є не лише технічним посередником між користувачем і функціональністю програми, а й важливою складовою, що формує загальне враження від роботи з системою, забезпечуючи інтуїтивність, адаптивність та візуальну цілісність рішення.

3.7 Тестування системи та приклади роботи

Процес тестування програмного забезпечення є критичним етапом у життєвому циклі системи, що забезпечує виявлення недоліків, перевірку коректності функціонування та підтвердження відповідності реалізованого функціоналу до вимог технічного завдання. У рамках цієї кваліфікаційної

роботи тестування проводилося з акцентом на функціональну стабільність системи, точність побудови рекомендацій, зручність інтерфейсу та коректність роботи з базою даних.

Функціональне тестування

На етапі функціонального тестування увагу було зосереджено на перевірці роботи ключових модулів програмного забезпечення в межах передбачених сценаріїв використання. Завдання полягало не лише в констатації коректності виконання окремих функцій, але й у з'ясуванні того, наскільки узгоджено взаємодіють між собою окремі компоненти системи.

Так, у процесі перевірки процедури реєстрації було встановлено, що система здатна успішно додавати нові облікові записи користувачів до бази даних. Крім того, алгоритм верифікації запобігає дублюванню імен, забезпечуючи унікальність кожного облікового запису. Це свідчить про дотримання базових принципів цілісності даних ще на етапі введення.

Механізм авторизації також продемонстрував стабільну роботу: під час тестування була підтверджена його здатність ідентифікувати зареєстрованого користувача за заданими обліковими даними та здійснювати перехід до основної форми застосунку без затримок або помилок у навігації.

Ще один ключовий аспект – це завантаження продукції з бази даних. Програма коректно виконує зчитування записів, що містять назви, ціни та шляхи до зображень товарів, після чого динамічно генерує візуальні картки для кожної позиції. Такий підхід забезпечує актуальність відображення вмісту магазину, що особливо важливо в умовах змінного асортименту.

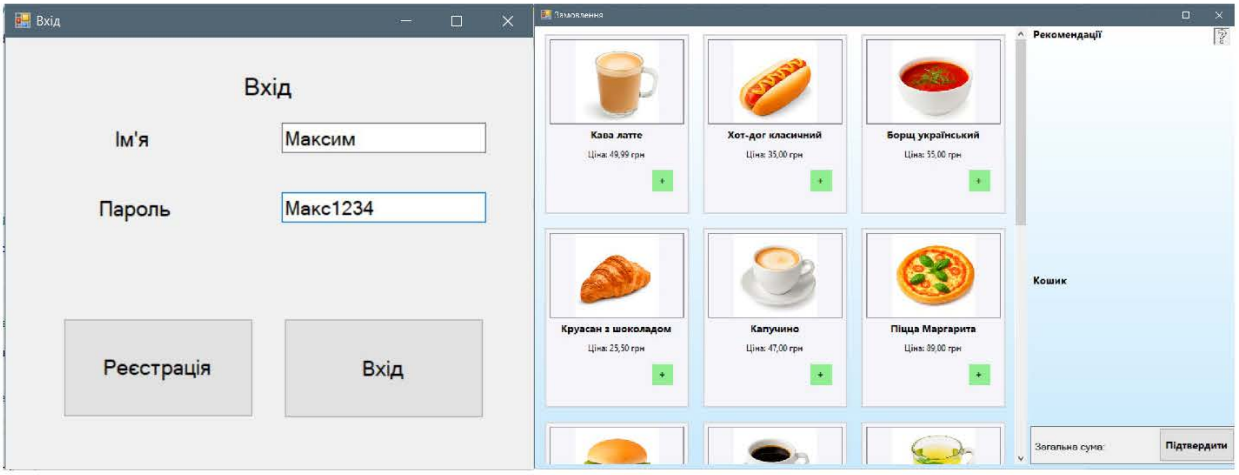

Рисунок 3.11-3.12 Вхід.

Робота кошика.

Було перевірено правильність додавання товарів, динамічне оновлення кількості, перерахунок суми та можливість видалення позицій. Всі операції відображаються миттєво, без необхідності перезавантаження інтерфейсу. Це свідчить про адекватну реакцію інтерфейсу на дії користувача та ефективну інтеграцію з логікою обробки даних.

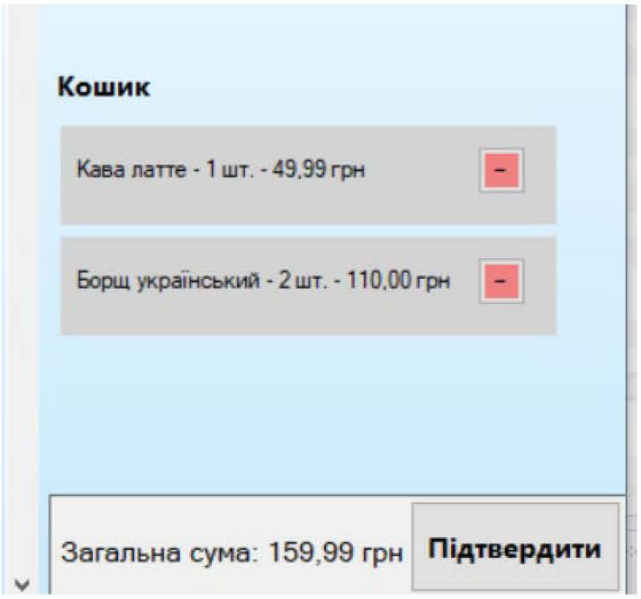

Рисунок 3.13 Кошик

Тестування блоку рекомендацій.

Під час взаємодії з кошиком система формує персоналізовані рекомендації. Було перевірено, що після додавання певних товарів до кошика алгоритм правильно знаходить замовлення з історії, що містять аналогічні набори, і на основі частотності пропонує релевантні додаткові товари.

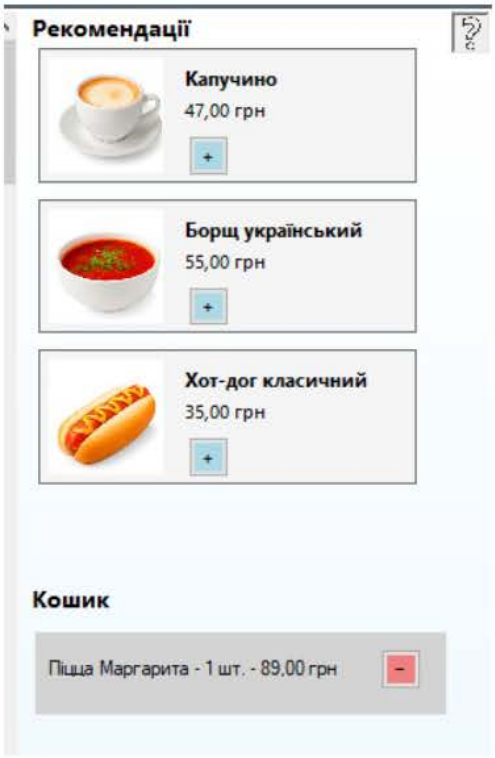

Рисунок 3.14 Рекомендації

Обробка виключень та нештатних ситуацій

Надійність програмного забезпечення значною мірою визначається не лише його здатністю функціонувати у стандартних умовах, а й тим, як система реагує на непередбачувані або нестандартні сценарії взаємодії з користувачем. У процесі тестування 0особлива увага приділялась імітації саме таких ситуацій, щоб перевірити рівень стійкості додатку до можливих помилок та відхилень від типового користувацького поведінкового патерну.

Одним із поширених сценаріїв є взаємодія з порожнім кошиком. Було перевірено, що при спробі підтвердити замовлення, не додавши жодного товару, система не лише блокує подальші дії, а й відображає чітке інформативне повідомлення, яке вказує на необхідність додати хоча б один

товар. Такий механізм дозволяє уникнути помилкових транзакцій та забезпечує логічну послідовність дій у рамках користувацького сценарію.

Інший важливий аспект пов’язаний із системою персоналізованих рекомендацій. У випадку, коли історичні дані не містять жодних релевантних поєднань товарів (тобто немає жодного замовлення з комбінацією, аналогічною поточному кошику), система не залишає користувача без пропозицій. Натомість, динамічно формується блок із найпопулярнішими товарами серед інших користувачів. Таким чином, реалізовано своєрідний механізм "резервного рівня", який зберігає функціональність системи навіть за відсутності повноцінних асоціативних зв’язків.

Також важливим елементом є реалізований захист від несанкціонованого доступу. При спробі увійти до системи з некоректними обліковими даними користувач отримує відповідне повідомлення про помилку, а доступ до внутрішніх функцій додатку блокується. Ця перевірка виконується на рівні бази даних і супроводжується чіткою візуалізацією результату для користувача.

Таким чином, система демонструє високий рівень готовності до роботи в умовах, коли поведінка користувача виходить за межі стандартного сценарію, що є важливим критерієм її надійності та практичної придатності.

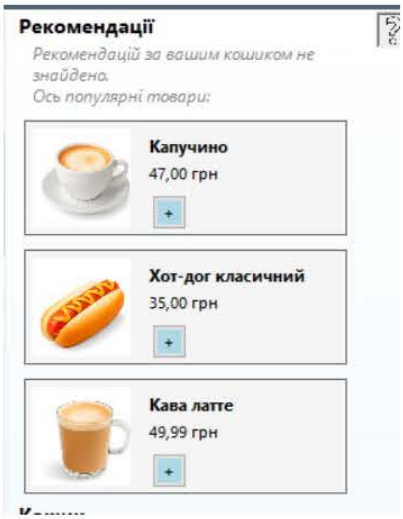

Рисунок 3.15 Найпопулярніші товари

Тестування запису до бази даних.

Було перевірено, що після підтвердження замовлення дані коректно зберігаються в таблиці Orders. Вміст таблиці відображає правильний набір товарів із кількістю та прив'язкою до користувача. Повторні замовлення інших користувачів також були враховані при формуванні нових рекомендацій, що підтверджує коректність механізму зчитування та запису транзакцій.

14	1	Борщ українсь...	02.05.2025 11:4...
15	1	Борщ українсь...	02.05.2025 11:5...
16	1	Борщ українсь...	02.05.2025 12:3...
17	1	Хот-дог класич...	02.05.2025 13:1...
18	1	Хот-дог класич...	02.05.2025 13:2...
19	1	Суп курячий×2...	02.05.2025 15:5...
20	1	Борщ українсь...	02.05.2025 16:4...
21	1	Кава латте×1, ...	02.05.2025 17:2...
22	3	Хот-дог класич...	05.05.2025 20:5...
23	3	Борщ українсь...	09.05.2025 12:2...
24	3	Борщ українсь...	09.05.2025 12:2...
25	3	Борщ українсь...	09.05.2025 12:2...
26	3	Піцца Маргар...	09.05.2025 12:2...
27	3	Борщ українсь...	09.05.2025 12:2...

Рисунок 3.16 Таблиця Orders

Додаткове тестування таблиці частотних зв'язків.

Функція, що дозволяє переглянути товари, які найчастіше з'являлись разом із поточним вмістом кошика, також була перевірена. Було встановлено, що таблиця коректно відображає дані лише у випадках повного збігу набору, ігноруючи власну історію користувача.

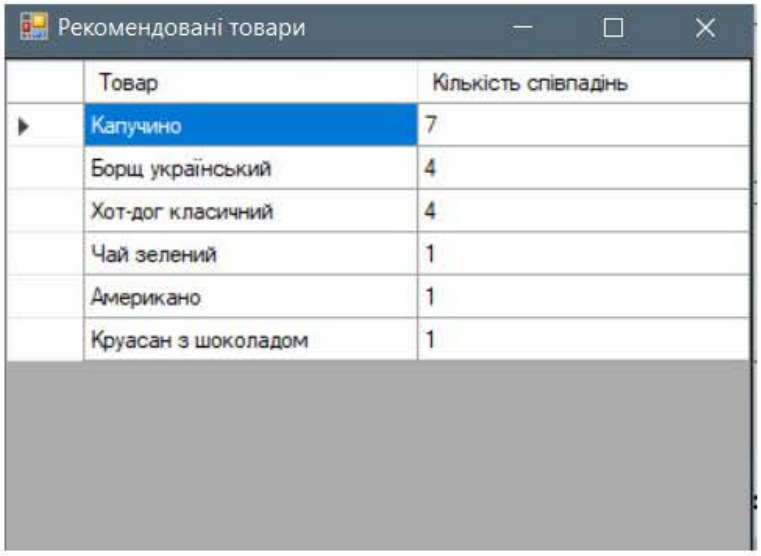

Рисунок 3.17 таблиці частотних зв'язків

Оцінка продуктивності.

Було відзначено, що система працює стабільно навіть при збільшенні обсягу історичних даних. Алгоритм пошуку рекомендацій оптимізовано за рахунок кешування та обмеження глибини пошуку, що дозволяє зберігати швидкодію при великій кількості записів.

Проведене тестування підтвердило надійність та стабільність функціонування системи в реальних умовах експлуатації. Усі функціональні елементи успішно пройшли перевірку, що свідчить про готовність системи до впровадження у комерційне середовище. Гнучкий підхід до формування рекомендацій, зручність інтерфейсу та ефективна робота з базою даних забезпечують високий рівень користувацького досвіду.

Висновки до розділу 3

У третьому розділі було здійснено безпосередню реалізацію рекомендаційної системи, що охоплює проектування архітектури, розробку бази даних, створення логіки побудови рекомендацій та дизайн інтерфейсу користувача. Побудовано локальну програму на платформі C# (WinForms), яка

функціонує без підключення до інтернету, що відповідає вимогам автономності для малих підприємств.

Створено реляційну структуру бази даних, яка включає таблиці користувачів, товарів та замовлень, з акцентом на гнучке зберігання транзакцій у текстовому форматі, що спрощує обробку на стороні застосунку. Реалізовано алгоритм формування рекомендацій, який оперативно аналізує в режимі реального часу вміст кошика користувача, знаходить релевантні транзакції в історії та виводить найбільш ймовірні супутні товари.

Окрему увагу приділено розробці візуального інтерфейсу: реалізовано динамічне оновлення рекомендацій, інтерактивний кошик, механізм перегляду частотних зв'язків між товарами. Проведено тестування функціоналу, що засвідчило стабільність системи, її адаптивність до нових даних і відповідність очікуванням користувача.

Узагальнюючи, третій розділ демонструє завершену реалізацію усіх функціональних компонентів системи — від проектування до практичного тестування — та доводить життєздатність обраної концепції у прикладному середовищі.

ВИСНОВОК

У рамках виконання кваліфікаційної роботи було здійснено повноцінне дослідження, розробку та практичну реалізацію рекомендаційної системи, орієнтованої на потреби компаній, що функціонують у сфері роздрібної торгівлі. Враховуючи зростання ролі персоналізованого підходу у взаємодії з клієнтами, особливо в умовах високої конкуренції, обрана тема продемонструвала як актуальність, так і значний практичний потенціал.

Першим кроком у межах дослідження було здійснено аналіз предметної області. Було визначено ключові особливості сучасної роздрібної торгівлі, де дедалі більшу роль відіграють цифрові інструменти взаємодії з клієнтом. Розглянуто характерні проблеми персоналізації, серед яких: інформаційне перевантаження, відсутність контекстної релевантності та складність у виявленні стабільних шаблонів споживчої поведінки без залучення аналітичних методів. Ці аспекти лягли в основу формулювання задачі, яка полягала у створенні зручної, інтерактивної та гнучкої системи рекомендацій.

У другому розділі здійснено порівняльний огляд існуючих підходів до побудови систем рекомендацій. Було проаналізовано алгоритми колаборативної фільтрації, контентного фільтрування, гібридні моделі, а також алгоритми асоціативного аналізу. На основі критичного аналізу, а також з огляду на специфіку доступних даних (історія транзакцій у вигляді наборів куплених товарів), було прийнято рішення використати методику асоціативних правил, що ґрунтується на виявленні частих співпадінь між товарами у замовленнях.

Реалізацію системи проведено за допомогою середовища C# (WinForms), з використанням бази даних SQL Server LocalDB. Було розроблено повноцінну клієнтську програму, що підтримує функції реєстрації, авторизації, додавання товарів у кошик, підтвердження замовлення, а також автоматичне формування рекомендацій.

Однією з ключових функціональних особливостей розробленої програми стала реалізація алгоритму, що імітує поведінку методу Apriori для побудови асоціативних рекомендацій. Після аналізу історії замовлень інших користувачів система виявляє товари, які найчастіше зустрічаються разом із тими, що містяться в поточному кошику. У випадку, коли релевантних співпадінь не виявлено, активується механізм підстановки найбільш популярних товарів. Такий підхід забезпечив гнучкість системи, дозволяючи генерувати рекомендації навіть при обмеженій кількості історичних даних.

Для візуалізації взаємозв'язків між товарами, які найчастіше зустрічаються у спільних замовленнях, було реалізовано інструмент перегляду статистики співпадінь. Це дозволяє користувачу краще зрозуміти логіку рекомендаційної системи, що сприяє підвищенню довіри до її рішень.

У межах тестування програма успішно пройшла перевірку за кількома сценаріями: додавання товарів до кошика, автоматичне формування рекомендацій, підтвердження замовлення з подальшим записом у базу даних, а також обробка виключень, зокрема ситуацій відсутності даних або некоректного введення.

Загалом, отримані результати свідчать про доцільність застосування алгоритмів інтелектуального аналізу даних у практичних завданнях бізнес-аналітики. Створена система може бути адаптована до реальних умов малих і середніх торгових підприємств, оскільки не потребує великих обсягів вхідних даних і може ефективно функціонувати на базі звичайної транзакційної інформації.

У перспективі можливим напрямком подальшого вдосконалення може стати розширення системи за рахунок глибшого профілювання користувача, введення часових і контекстних факторів, а також інтеграція з веб-платформами чи мобільними застосунками. У такому вигляді розроблена система може слугувати базовим прикладом впровадження персоналізованої торгівлі на практиці.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Андрущенко В. П., Ільченко І. І. Основи інформатики та комп'ютерної техніки. Київ : Либідь, 2020. 376 с.

2. Бойко А. О. Програмування мовою C# у середовищі Visual Studio. Харків : Фоліо, 2021. 284 с.

3. Воробйов Є. В., Петренко Ю. С. Бази даних : навч. посібник. Дніпро : УМСФ, 2022. 176 с.

4. Гаврилюк О. О. Основи створення інтерфейсів користувача : метод. вказівки. Львів : ЛНУ ім. І. Франка, 2021. 94 с.

5. ДСТУ 8302:2015. Інформація та документація. Бібліографічне посилання. Загальні положення та правила складання. Київ : Мінекономрозвитку України, 2016. 16 с.

6. Златопольський Д. В. Мови програмування: C#, Python, Java : підручник. Київ : Освіта України, 2020. 412 с.

7. IT-компанії: тенденції розвитку та впровадження штучного інтелекту в ритейлі [Електронний ресурс] // AIN.ua. URL: https://ain.ua (дата звернення: 26 травня 2025 р.).

8. Ковальчук А. С. Методи штучного інтелекту в електронній комерції: теорія і практика. Київ : КНЕУ, 2023. 312 с.

9. Кузьмін О. Є., Мельник І. Г. Інтелектуальний аналіз даних : підручник. Львів : ЛНУ, 2019. 234 с.

10. Литвиненко С. П. Рекомендаційні системи: структура, класифікація, алгоритми // Наука й інновації. 2021. № 3(17). С. 41–52.

11. Митрофанова І. В. Інтелектуальні інформаційні системи. Київ : Наука, 2018. 328 с.

12. Офіційний сайт Microsoft Docs [Електронний ресурс]. URL: https://docs.microsoft.com/ (дата звернення: 26 травня 2025 р.).

13. Панкратов Ф. М. Бази даних у середовищі SQL Server : навч. посібник. Дніпро : УМСФ, 2021. 140 с.

14. Савченко Л. В. Розробка інтерфейсу користувача у Windows Forms. Харків : ХНУРЕ, 2020. 116 с.

15. Шапошніков С. М. Основи штучного інтелекту. Одеса : ОНУ ім. І. Мечникова, 2022. 298 с.

16. Aggarwal C. C. Recommender Systems: The Textbook. Cham : Springer, 2016. 493 p.

17. Adomavicius G., Tuzhilin A. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions // IEEE Transactions on Knowledge and Data Engineering. 2005. Vol. 17, No. 6. P. 734–749.

18. Burke R. Hybrid Recommender Systems: Survey and Experiments // User Modeling and User-Adapted Interaction. 2002. Vol. 12. P. 331–370.

19. Han J., Pei J., Kamber M. Data Mining: Concepts and Techniques. 3rd ed. Burlington : Morgan Kaufmann, 2011. 744 p.

20. He X., Liao L., Zhang H., Nie L., Hu X., Chua T. S. Neural Collaborative Filtering // Proceedings of the 26th International Conference on World Wide Web. 2017. P. 173–182.

21. Jannach D., Adomavicius G., Tuzhilin A. Recommender Systems – Challenges, Frontiers and Applications // Machine Learning. 2021. Vol. 110. P. 1937–1969.

22. Koren Y., Bell R., Volinsky C. Matrix Factorization Techniques for Recommender Systems // Computer. 2009. Vol. 42, No. 8. P. 30–37.

23. Linden G., Smith B., York J. Amazon.com Recommendations: Item-to-Item Collaborative Filtering // IEEE Internet Computing. 2003. Vol. 7, No. 1. P. 76–80.

24. Ricci F., Rokach L., Shapira B. Introduction to Recommender Systems Handbook. Cham : Springer, 2011. 845 p.

25. Schafer J. B., Frankowski D., Herlocker J., Sen S. Collaborative Filtering Recommender Systems // The Adaptive Web. Berlin : Springer, 2007. P. 291–324.

26. Tan P.-N., Steinbach M., Kumar V. Introduction to Data Mining. Harlow : Pearson, 2019. 864 p.

27. Zhang Y., Chen X. Explainable Recommendation: A Survey and New Perspectives // Foundations and Trends in Information Retrieval. 2020. Vol. 14, No. 1. P. 1–101.

28. Zhou T., Kuscsik Z., Liu J.-G., Medo M., Wakeling J. R., Zhang Y.-C. Solving the apparent diversity-accuracy dilemma of recommender systems // Proceedings of the National Academy of Sciences. 2010. Vol. 107, No. 10. P. 4511–4515.

29. Федорчук П. М. Методи інтелектуального аналізу даних : навч. посібник. Київ : НТУУ «КПІ», 2020. 204 с.

30. Шапаренко О. М. Моделювання користувацьких уподобань у рекомендаційних системах. Харків : ХНУРЕ, 2021. 168 с.

Додаток А

Лістинг основного алгоритму

```
    private void UpdateRecommendations()
{
    HashSet<string> currentProducts = cart.Keys.ToHashSet();
    Dictionary<string, int> frequency = new Dictionary<string, int>();
    flowLayoutPanelRecommendations.Controls.Clear();
    if (currentProducts.Count == 0)
        return;
    int relevantTransactionCount = 0;
    double Supp_min = 0.2;

    using (SqlConnection conn = new SqlConnection(connectionString))
    {
        conn.Open();
        string query = @"
    SELECT Products FROM Orders
    WHERE UserId != @userId";
        SqlCommand cmd = new SqlCommand(query, conn);
        cmd.Parameters.AddWithValue("@userId", Form1.CurrentUserId);
        SqlDataReader reader = cmd.ExecuteReader();
        while (reader.Read())
        {
            string productsField = reader["Products"].ToString();
            string[] items = productsField.Split(',');
            HashSet<string> orderProducts = new HashSet<string>();

            foreach (string item in items)
            {
                string name = item.Trim().Split('×')[0];
                orderProducts.Add(name);
            }
            if (currentProducts.All(p => orderProducts.Contains(p)))
            {
                relevantTransactionCount++;

                foreach (string product in orderProducts)
                {
                    if (!currentProducts.Contains(product))
                    {
                        if (frequency.ContainsKey(product))
                            frequency[product]++;
```

```
                    else
                        frequency[product] = 1;
                }
            }
        }
    }
    reader.Close();
}

if (frequency.Count == 0 || relevantTransactionCount == 0)
{
    Label noMatchLabel = new Label();
    noMatchLabel.Text = "Рекомендацій за вашим кошиком не
знайдено.\nОсь популярні товари:";
    noMatchLabel.ForeColor = Color.Gray;
    noMatchLabel.Font = new Font("Segoe UI", 9, FontStyle.Italic);
    noMatchLabel.AutoSize = true;
    noMatchLabel.Padding = new Padding(5);
    flowLayoutPanelRecommendations.Controls.Add(noMatchLabel);
    ShowPopularRecommendations();
    return;
}
var support = frequency.ToDictionary(
    kv => kv.Key,
    kv => (double)kv.Value / relevantTransactionCount
);
var topProducts = support
    .Where(p => p.Value >= Supp_min) // ❖ фільтрація за порогом
    .OrderByDescending(p => p.Value)
    .Take(3)
    .Select(p => p.Key);
if (!topProducts.Any())
{
    Label noMatchLabel = new Label();
    noMatchLabel.Text = "Жоден товар не перевищує поріг
підтримки.\nОсь популярні товари:";
    noMatchLabel.ForeColor = Color.Gray;
    noMatchLabel.Font = new Font("Segoe UI", 9, FontStyle.Italic);
    noMatchLabel.AutoSize = true;
    noMatchLabel.Padding = new Padding(5);
    flowLayoutPanelRecommendations.Controls.Add(noMatchLabel);
    ShowPopularRecommendations();
    return;
```

```
    }
    foreach (string recommendedProduct in topProducts)
    {
        AddRecommendationCard(recommendedProduct);
    }
}
```

Додаток Б

Метод формування загальних (неперсоналізованих) рекомендацій

```
private void ShowPopularRecommendations()
{
    Dictionary<string, int> frequency = new Dictionary<string, int>();
    using (SqlConnection conn = new SqlConnection(connectionString))
    {
        conn.Open();
        string query = "SELECT Products FROM Orders WHERE UserId != 
@userId";
        SqlCommand cmd = new SqlCommand(query, conn);
        cmd.Parameters.AddWithValue("@userId", Form1.CurrentUserId);
        SqlDataReader reader = cmd.ExecuteReader();
        while (reader.Read())
        {
            string[] items = reader["Products"].ToString().Split(',');
            foreach (string item in items)
            {
                string name = item.Trim().Split('×')[0];

                if (cart.ContainsKey(name)) continue; // не радимо те, що вже в 
кошику

                if (frequency.ContainsKey(name))
                    frequency[name]++;
                else
                    frequency[name] = 1;
            }
        }
        reader.Close();
    }
    var topProducts = frequency.OrderByDescending(p => p.Value)
                               .Take(3)
                               .Select(p => p.Key);
    foreach (string product in topProducts)
    {
        AddRecommendationCard(product);
    }
}
```

Додаток В

Метод відображення частот товарних асоціацій

```
private void ShowProductAssociationFrequencies()
{
    if (cart.Count == 0)
    {
        MessageBox.Show("Кошик порожній.");
        return;
    }
    HashSet<string> cartSet = new HashSet<string>(cart.Keys);
    Dictionary<string, int> productFrequencies = new Dictionary<string,
int>();
    int relevantTransactionCount = 0; // нова змінна для підтримки

    using (SqlConnection conn = new SqlConnection(connectionString))
    {
        conn.Open();
        string query = "SELECT Products FROM Orders WHERE UserId !=
@userId";
        SqlCommand cmd = new SqlCommand(query, conn);
        cmd.Parameters.AddWithValue("@userId", Form1.CurrentUserId);
        SqlDataReader reader = cmd.ExecuteReader();
        while (reader.Read())
        {
            string[] productsRaw = reader["Products"].ToString().Split(',');
            List<string> products = productsRaw
                .Select(p => p.Split('×')[0].Trim())
                .Where(p => !string.IsNullOrWhiteSpace(p))
                .ToList();
            HashSet<string> transactionSet = new HashSet<string>(products);

            if (cartSet.All(item => transactionSet.Contains(item)))
            {
                relevantTransactionCount++; // рахуємо релевантні транзакції

                foreach (string product in products)
                {
                    if (!cartSet.Contains(product))
                    {
                        if (productFrequencies.ContainsKey(product))
                            productFrequencies[product]++;
                        else
```

```
                        productFrequencies[product] = 1;
                    }
                }
            }
        }
    }

    if (productFrequencies.Count == 0 || relevantTransactionCount == 0)
    {
        MessageBox.Show("Немає відповідних комбінацій у історії.");
        return;
    }
    var support = productFrequencies.ToDictionary(
        kv => kv.Key,
        kv => (double)kv.Value / relevantTransactionCount
    );
    Form resultForm = new Form();
    resultForm.Text = "Рекомендовані товари";
    resultForm.Size = new Size(500, 300);
    DataGridView dgv = new DataGridView();
    dgv.Dock = DockStyle.Fill;
    dgv.ReadOnly = true;
    dgv.AllowUserToAddRows = false;
    dgv.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;
    dgv.Columns.Add("Product", "Товар");
    dgv.Columns.Add("Count", "Кількість співпадінь");
    dgv.Columns.Add("Support", "Підтримка");
    foreach (var pair in productFrequencies.OrderByDescending(p =>
p.Value))
    {
        double supportValue = support.ContainsKey(pair.Key) ?
support[pair.Key] : 0;
        dgv.Rows.Add(pair.Key, pair.Value, supportValue.ToString("P2")); //
формат як відсоток
    }

    resultForm.Controls.Add(dgv);
    resultForm.ShowDialog();
}       }
    if (productFrequencies.Count == 0)
    {
        MessageBox.Show("Немає відповідних комбінацій у історії.");
        return;
```

```
        }
        Form resultForm = new Form();
        resultForm.Text = "Рекомендовані товари";
        resultForm.Size = new Size(400, 300);
        DataGridView dgv = new DataGridView();
        dgv.Dock = DockStyle.Fill;
        dgv.ReadOnly = true;
        dgv.AllowUserToAddRows = false;
        dgv.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;
        dgv.Columns.Add("Product", "Товар");
        dgv.Columns.Add("Count", "Кількість співпадінь");
        foreach (var pair in productFrequencies.OrderByDescending(p =>
p.Value))
        {
            dgv.Rows.Add(pair.Key, pair.Value);
        }
        resultForm.Controls.Add(dgv);
        resultForm.ShowDialog();
    }
```