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)
«__2_» __червня_____ 2025р.	«_2__» ____червня_____ 2025р.

КВАЛІФІКАЦІЙНА РОБОТА
НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ БАКАЛАВРА

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

освітньо-професійної програми _Інформаційні системи та штучний інтелект___

на тему: Розроблення клієнтського мобільного додатку мережі магазинів "Смак Кореї"

Виконав: здобувач _4_ курсу, групи __КН-4-4___

_____Заварзін Андрій Олександрович_____ _____
(прізвище, ім'я по батькові повністю) (підпис)

Керівник _Харкянен Олена Валеріївна_____ _____
(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 _____ _____
(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 – 2025 р.

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ

Інститут (факультет) Автоматизації і комп'ютерних систем
Кафедра _інформаційних технологій, штучного інтелекту і кібербезпеки ___
Освітній ступінь бакалавр
Спеціальність 122 «Комп'ютерні науки»
Освітньо-професійна програма __Інформаційні системи та штучний інтелект

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інформаційних технологій, штучного інтелекту і кібербезпеки _____
_____ Сергій ГРИБКОВ

«_28_» _____квітня____ 2025 року

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧА
Заварзін Андрій Олександрович
(прізвище, ім'я, по батькові)

1. Тема роботи Розроблення клієнтського мобільного додатку мережі магазинів "Смак Кореї"
керівник роботи Харкянен Олена Валеріївна, к.т.н., доцент,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом закладу вищої освіти від 28 квітня 2025 року № 254 кс
2. Строк подання здобувачем роботи: 30.05.2025 р.
3. Вихідні дані до роботи: асортимент товарів магазину «Смак Кореї», шаблони нормативних документів, чеки, звітна документація.
4. Зміст пояснювальної записки (перелік питань, які потрібно розробити):
1. Загальна характеристика ТОВ «Відус»
3. Розробка функціональної моделі
4. Огляд систем-аналогів для розв'язання виявлених проблем
6. Розрахунок економічного ефекту від впровадження мобільного доатку
7. Постановка задачі
8. Проектування бази даних
9. Реалізація комплексу задач автоматизаці
10. Інструкція користувача
11. Тестування програмного продукту
5. Перелік графічного матеріалу:
Моделі бізнес-процесів у нотації BPMN, логічна модель БД, фізична модель БД, модель бази даних в MS SQL Server, кадри інтерфейсу користувача

6. Консультанти розділів роботи:

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Харкянен О.В., доцент НУХТ		
2	Харкянен О.В., доцент НУХТ		
3	Харкянен О.В., доцент НУХТ		

7. Дата видачі завдання: 28 квітня 2025 року

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів виконання кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Дослідження предметної області	28.04.25-30.04.25	Виконано
2	Розробка функціональної моделі	01.05.25-03.05.25	Виконано
3	Проектування БД	04.05.25-06.05.25	Виконано
4	Розробка технічного завдання	07.05.25-09.05.25	Виконано
5	Визначення вимог до функцій системи	10.05.25-12.05.25	Виконано
6	Реалізація задач автоматизації системи	13.05.25-20.05.25	Виконано
7	Оформлення пояснювальної записки	21.05.25-25.05.25	Виконано
8	Створення презентації	26.05.25-30.05.25	Виконано

Здобувач _____ Заварзін А.О.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Харкянен О. В.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Кваліфікаційна робота на тему «Розробка мобільного додатку для мережі магазинів “Смак Кореї”» присвячена дослідженню процесу взаємодії продуктової мережі магазинів роздрібної торгівлі з покупцями та розробці мобільного застосунку для покращення їх обслуговування. У роботі проаналізовано сучасний стан цифрових сервісів компанії, виявлено обмеження відсутності мобільного рішення, досліджено приклади конкурентів, а також обґрунтовано потребу у створенні власного зручного застосунку з підтримкою бонусної програми.

Розроблено повноцінний клієнтський мобільний додаток із використанням .NET MAUI, ASP.NET Core Web API та Microsoft SQL Server. Застосунок реалізує ключові функції: реєстрацію та авторизацію користувача, перегляд товарів, фільтрацію, керування кошиком, оформлення замовлень, облік бонусів, а також перегляд історії покупок і даних профілю.

Проведено техніко-економічне обґрунтування розробки, яке підтвердило ефективність впровадження — очікуваний термін окупності становить 1,6 року, економічний ефект складає 760 000 грн.

Кваліфікаційна робота обсягом 70 сторінок містить 29 ілюстрацій, 7 таблиць, 29 використаних джерел та 5 додатків.

Ключові слова: МОБІЛЬНИЙ ЗАСТОСУНОК, РОЗДРІБНА ТОРГІВЛЯ, БОНУСНА ПРОГРАМА, ІНФОРМАЦІЙНА СИСТЕМА, .NET MAUI.

SUMMARY

The qualification work on the topic ‘Development of a mobile application for the Taste of Korea chain of stores’ is devoted to the study of the process of interaction between the grocery chain of retail stores and customers and the development of a mobile application to improve their service. The paper analyses the current state of the company's digital services, identifies the limitations of the lack of a mobile solution, studies examples of competitors, and justifies the need to create its own convenient application with support for a bonus programme.

A full-fledged client mobile application was developed using .NET MAUI, ASP.NET Core Web API and Microsoft SQL Server. The application implements key functions such as user registration and authorisation, product browsing, filtering, basket management, ordering, bonus accounting, as well as viewing purchase history and profile data.

A feasibility study was carried out, which confirmed the effectiveness of the implementation: the expected payback period is 1.6 years, and the economic effect is UAH 760,000.

The 70-page qualification work contains 29 illustrations, 7 tables, 29 references and 5 appendices.

Keywords: MOBILE APPLICATION, RETAIL, BONUS PROGRAMME, INFORMATION SYSTEM, .NET MAUI.

ЗМІСТ

ВСТУП

В умовах активного розвитку цифрових технологій у сфері роздрібної торгівлі особливого значення набуває створення зручних та функціональних мобільних застосунків для взаємодії з клієнтами. Сучасні покупці очікують швидкого доступу до сервісів із будь-якого мобільного пристрою, стабільної роботи додатку та зрозумілого інтерфейсу. У зв’язку з цим розробка мобільного застосунку для мережі магазинів «Смак Кореї» є актуальною та доцільною, адже вона сприяє покращенню взаємодії з клієнтами, підвищує їх лояльність та оптимізує процеси електронного замовлення продукції.

Метою цієї кваліфікаційної роботи є створення повноцінного клієнтського мобільного застосунку для «Смаку Кореї» з використанням технології .NET MAUI для клієнтської частини, ASP.NET Core Web API для серверної логіки та Microsoft SQL Server для зберігання даних. Основну увагу приділено автоматизації таких процесів, як реєстрація та авторизація користувачів, перегляд каталогу товарів, формування замовлень, облік бонусів і керування профілем.

У межах роботи було проведено аналіз предметної області, визначено функціональні вимоги до системи, сформульовано технічне завдання, спроєктовано архітектуру застосунку, реалізовано ключовий функціонал та проведено тестування.

Результатом проєкту є готовий до впровадження мобільний додаток, що відповідає сучасним вимогам щодо продуктивності, зручності використання та надійності, а також підтримує можливість масштабування для подальшого розвитку функціоналу.

РОЗДІЛ 1. ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1. Загальна характеристика ТОВ «ВІДУС»

1.1.1. Загальна інформація про ТОВ «ВІДУС»

ТОВ «ВІДУС» є локально відомим імпортером корейських товарів в Україну, забезпечуючи постачання від виробника напряму до кінцевого споживача. Компанія має два ключові напрями діяльності: оптові поставки для спеціалізованих магазинів і ресторанів, що є основним способом доходу, та розвиток власної мережі роздрібних магазинів корейських товарів.

Прямі контракти з виробниками дозволяють мати оптимальні ціни на товари та забезпечувати високу якість продукції для українського ринку. Компанія постійно розширює асортимент, приймаючи до уваги зростаючий попит на корейську продукцію та тренди.

Особлива увага приділяється контролю якості відповідно до міжнародних стандартів HACCP та ISO. Сучасні технології доставки забезпечують вчасне транпортування товару у належних умовах зберігання. Завдяки відповідальному підходу до організації поставок та контролю якості, ТОВ «ВІДУС» гарантує своїм клієнтам швидкий сервіс та високоякісну продукцію, напряму з країни виробника.

Підприємство спеціалізується в основному на логістичному обслуговуванні та має такі основні напрями діяльності:

1. Транспортне оброблення вантажів (КВЕД 52.24) - забезпечення повного циклу вантажних операцій.
2. Надання послуг перевезення речей (КВЕД 49.42) - організація комплексних переміщень майна.
3. Вантажний автомобільний транспорт (КВЕД 49.41) - здійснення автомобільних вантажоперевезень.
4. Інша допоміжна діяльність у сфері транспорту (КВЕД 52.29) - забезпечення супутніх транспортних послуг.

5. Складське господарство (КВЕД 52.10) - надання послуг зі зберігання товарів.

1.1.2. Мета та основні цілі магазину

ТОВ «ВІДУС» спеціалізується на імпорті товарів із Південної Кореї та має суттєві переваги з-поміж своїх конкурентів. Перш за все, це налагоджена довгими та надійними стосунками пряма взаємодія з корейськими постачальниками, що забезпечує постачання продукції безпосередньо від виробників, без можливості підробки продуктів. Такий формат дозволяє уникати додаткових витрат на посередницькі послуги, пропонуючи клієнтам доступні ціни.

Ще однією із ключових переваг є оптимізована логістична інфраструктура. Компанія застосовує сучасні транспортні рішення, що сприяє скороченню часу доставки й забезпечує належне зберігання продукції.

Додатково, компанія досліджує тренди та запити клієнтів, що дозволяє підвищити популярність магазину та прибуток.

Головною метою компанії є постачання на український ринок якісною імпортною продукцією Південної Кореї. ТОВ «ВІДУС» прагне сформувати стабільну систему постачання, яка ефективно відповідає потребам клієнтів, або спеціалізованих закладів, таких як ресторани та спеціалізовані торговельні точки.

Ключовим завданням є розширення власної мережі магазинів і розширення клієнтської бази. Компанія планує зосередитися на зміцненні та розвиненні партнерських відносин, удосконаленні логістики та створенні вигідних умов для співпраці покупцями.

Загалом, діяльність ТОВ «ВІДУС» спрямована на забезпечення високої якості обслуговування клієнтів, розширення асортименту та просування корейських товарів на українському ринку.

Основні види діяльності:

ТОВ «ВІДУС» здійснює комплексну діяльність у сфері логістики, транспортуванні та зберіганні товарів. Основні напрями роботи компанії охоплюють такі ключові сфери:

1. Транспортне оброблення вантажів (КВЕД 52.24)

Компанія надає послуги з перевезення, сортування та підготовки товарів до транспортування.

2. Вантажні автомобільні перевезення (КВЕД 49.41)

Компанія здійснює перевезення товарів автомобільним транспортом, використовуючи перевірених партнерів. Завдяки цьому постачання відбувається згідно встановлених термінів.

3. Допоміжна діяльність у сфері транспорту (КВЕД 52.29)

ТОВ «ВІДУС» також надає підтримку логістичних процесів, що включає оформлення перевезень, створення оптимального маршруту та контроль процесу доставки. Це сприяє ефективній роботі логістичних процесів і підвищенню ефективності транспортування продукції.

4. Складське господарство (КВЕД 52.10)

Компанія володіє високотехнологічними складськими приміщеннями, обладнаними відповідно до стандартів зберігання. Завдяки цьому товари зберігаються в правильних умовах, що забезпечує їх якість.

ТОВ "ВІДУС" забезпечує стабільне надходження товарів на український ринок та ретельно контролює процес — від відправлення виробником до продажу в магазинах та для імпорту до постачальників.

1.2. Організаційна структура магазину корейської продукції «Смак Кореї»

ТОВ «ВІДУС» було засновано у 2009 році як транспортно-логістична компанія, яка сьогодні представляє собою підприємство з двома ключовими напрямами діяльності: вантажними перевезеннями та роздрібною торгівлею корейськими товарами. У 2020 році компанія здійснила стратегічне розширення бізнесу, запустивши мережу магазинів «Смак Кореї», що спеціалізується на

продажу продуктів харчування та побутових товарів південнокорейського виробництва.

Компанія має централізовану систему управління з головним офісом у Києві, що забезпечує оптимальну координацію всіх бізнес-процесів. Ефективність діяльності підприємства базується на чітко структурованій організації, де кожен відділ виконує свої узгоджені функції. Логістичний підрозділ має основну роботу з товарами, імпорт до складського зберігання та доставки до точок продажу. Комерційний відділ зосереджується на розвитку роздрібної мережі та взаємодії з клієнтами, за допомогою маркетингових кампаній. Адміністративний сектор це юридичний департамент та служба безпеки, яка гарантує контроль та стабільне функціонування всієї організаційної структури.

1.2.1. Загальна схема організаційної структури

Для чіткої та ефективної роботи компанії було створено 4 відділи та розподіллено обов’язки між ними, згідно з потребами. Організаційна структура наведена на рисунку 1.1 і побудована таким чином, щоб забезпечити якісне обслуговування клієнтів, ефективно управляти персоналом і фінансами та рівномірно розподілити обов’язки між працівниками.

Рисунок 1.1 – Організаційна структура ТОВ «ВІДУС»

1. Директор

Директор є головним керівником, який формулює стратегічні цілі, ухвалює ключові рішення та координує роботу всіх підрозділів. Він контролює фінансову стратегію компанії, розвиває партнерські відносини та розширює ринки збуту корейських товарів.

2. Адміністративний відділ

Адміністративний відділ ТОВ "ВІДУС" забезпечує ефективне функціонування компанії через координацію управлінських процесів. Адміністратор контролює обіг документів, внутрішню комунікацію та організацію роботи. Відділ кадрів відповідає за знаходження персоналу, навчання, облік та підтримання дружніх стосунків із робітниками. Бухгалтерія веде фінансову звітність, контролює витрати та прибуток, підтримує економічну стабільність підприємства, сприяючи його розвитку.

3. Логістичний відділ

Логістичний відділ відповідає за регулярне та надійне постачання корейських товарів. Підрозділ імпорту товарів напряму підтримує контакт з постачальниками, заключаючи договори та контролюючи якість і правильність прибувших товарів. Складський підрозділ управляє зберіганням, обліком і підготовкою товарів до відвантаження. Транспортування забезпечує ефективні маршрути та швидку доставку до магазинів і партнерів.

4. Комерційний відділ

Комерційний відділ ТОВ "ВІДУС" відповідає за розвиток продажів і знаходження партнерів. Роздрібні продажі управляють діяльністю магазинів "Смак Кореї", залучаючи нових споживачів, підтримуючи гарний сервіс та дружні стосунки з постійними покупцями. Оптові продажі співпрацюють із ресторанами, гіпермаркетами, укладаючи контракти та розширюючи свій вплив. Маркетинг і SMM просувають бренд через рекламу та активність у соціальних мережах, підвищуючи впізнаваність компанії.

5. IT-відділ

IT-відділ підтримує технологічну сторону компанії. Системний адміністратор забезпечує стабільну роботу серверів, мереж і безпеки даних. Технічна підтримка вирішує технічні питання співробітників і клієнтів, зокрема щодо сайту та додатка "Смак Кореї". Спеціаліст касових систем управляє платіжними системами, касовими операціями й базою даних товарів.

1.2.2. Структура комерційного відділу

Головною метою комерційного відділу ТОВ "Відус", структуру якого наведеного на рисунку 1.2 є забезпечення зростання продажів корейських товарів, розширення партнерств і підвищення впізнаваності бренду "Смак Кореї". Відділ фокусується на ефективній реалізації продукції через роздрібні та оптові канали, залученні нових клієнтів і просуванні товарів на українському ринку, оптимізуючи процеси збуту для стабільного розвитку компанії.

Функції комерційного відділу:

1. Роздрібні продажі — управління діяльністю магазинів "Смак Кореї", консультація клієнтів, оформлення покупок і контроль якості обслуговування в торгових точках;
2. Оптові продажі — укладання контрактів із ресторанами, гіпермаркетами й дистриб’юторами, координація поставок великих партій корейських товарів;
3. Маркетинг і SMM — аналіз ринку, розробка рекламних кампаній, просування бренду в соціальних мережах, створення контенту для залучення покупців;
4. Розвиток партнерств — пошук нових клієнтів, оцінка попиту, підтримка довгострокових зв’язків із оптовими партнерами;
5. Аналіз продажів — моніторинг результатів, підготовка звітів про ефективність збуту, адаптація стратегій до ринкових змін;
6. Залучення клієнтів — організація акцій, знижок і програм лояльності для стимулювання попиту на корейську продукцію;

7. Координація з логістикою — забезпечення своєчасної доставки товарів до магазинів і партнерів для підтримки продажів.

Основні задачі комерційного відділу:

1. Збільшення продажів — підвищення обсягів реалізації корейських товарів у роздрібних магазинах "Смак Кореї" та через оптові канали;
2. Розширення партнерської мережі — пошук нових оптових клієнтів, зокрема ресторанів і гіпермаркетів, для розширення збуту;
3. Просування бренду — аналіз споживчих потреб, розробка маркетингових стратегій і активність у соціальних мережах для залучення аудиторії;
4. Оптимізація продажів — вдосконалення процесів у магазинах, контроль якості обслуговування й адаптація асортименту до попиту;
5. Управління акціями — організація знижок, програм лояльності й розпродажів для стимулювання покупок;
6. Координація з логістикою — забезпечення своєчасного постачання товарів до торгових точок і партнерів для безперебійних продажів;
7. Моніторинг ринку — аналіз конкурентів, оцінка тенденцій і коригування стратегій для підтримки конкурентоспроможності.

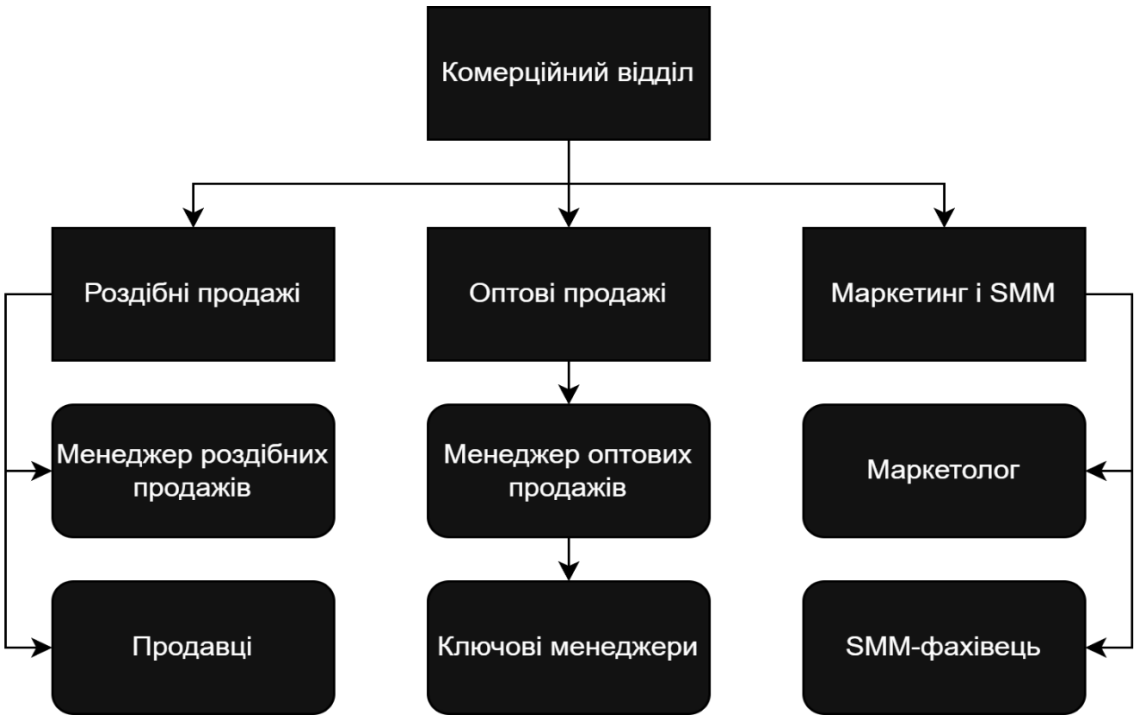

Рисунок 1.2 – Організаційно-виробнича структура комерційного відділу

Розглянемо посадові обов’язки робітників комерційного відділу, наведені у таблиці 1.1.

Таблиця 1.1. Посадові обов’язки робітників комерційного відділу

Працівники	**Функції**	**Взаємодія з іншими відділами**
Менеджер роздрібних продажів	• Координування діяльності магазинів «Смак Кореї». • Контроль продажів. • Забезпечення навчання продавців. • Оптимізація роботи торгових точок.	• Погодження кадрових рішень з адміністративним відділом. • Передання інформації про потреби магазинів у товарі в логістичний відділ. • Усунення технічних збоїв у POS-системах з IT-відділом.
Продавці	• Консультація клієнтів. • Оформлення покупок. • Контроль асортименту у магазинах. • Забезпечення високої якості обслуговування.	• Надання зворотного зв’язку щодо наявності товарів та передання заявок на поповнення залишків з логістичним відділом.
Менеджер оптових продажів	• Укладання контрактів з ресторанами, гіпермаркетами та дистриб’юторами. • Аналіз попиту. • Розвиток партнерських відносин. • Координація поставок корейської продукції.	• Узгодження умов контрактів з адміністративним відділом. • Формування замовлення для оптових клієнтів та контроль своєчасності відвантажень.
Ключові менеджери	• Робота з великими клієнтами, оцінюючи їх потреби.	• Погодження умов співпраці з ключовими клієнтами та звітність про виконання

Працівники	Функції	Взаємодія з іншими відділами
	• Забезпечення стабільних поставок. • Підвищення обсягів оптових продажів.	плану з адміністративним відділом. • Контроль стабільності поставок та узгодження термінів з логістичним відділом.
Маркетолог	• Виконання аналізу ринку. • Розробка рекламних кампаній. • Формування стратегії просування бренду «Смак Кореї».	• Розробка технічного завдання для промо, розсилок та кампаній з IT-відділом. • Погодження бюджету та отримання стратегічних планів від адміністративного відділку.
SMM-фахівець	• Просування кампанії в соціальних мережах. • Створення та публікація контенту. • Координація рекламної активності в інтернеті. • Залучення нової аудиторії.	• Взаємодія з технічними спеціалістами для інтеграції соціальних мереж за допомогою IT-відділу. • Отримання інформації про нові надходження для створення публікацій від логістичного відділу.

Для правильної роботи даного відділу була створена налагоджена система обміну документами, наведена у таблиці 1.2.

Таблиця 1.2. Інформаційна взаємодія комерційного відділу

Відділ	Одержані документи	Надані документи
Комерційний	• Заявки на товар від магазинів. • Комерційні пропозиції від постачальників. • Квитанції та ТТН від постачальників. • Звіти з аналізів продажів та ефективність акцій. • Планові показники продажів від адміністративного відділу	• Рахунки на оплату оптовим клієнтам. • Комерційні пропозиції та презентації для магазинів. • Звіти про продажі. • Інформація щодо товарів, що потребують просування.

1.3. Аналіз нинішнього стану комп'ютеризації ТОВ «ВІДУС»

1. Адміністративний відділ

Основна діяльність відділ здійснюється за допомогою стандартного офісного програмного забезпечення (Word, Excel, Google Диск). Документи ведуться частково в електронному вигляді, а частково у паперовому. На даному етапі жодної системи управління персоналом або внутрішніми процесами не впроваджено;

2. Логістичний відділ

Планування доставок, вибір маршрутів та переміщень товару виконується за допомогою електронних таблиць або месенджерів. Окремого програмного забезпечення для відслідковування транспорту не впроваджено;

3. ІТ-відділ

Займається базовою технічно підтримкою всієї системи, налаштуванням програм і обладнанням та оновленням сайту. ІТ-відділ не має повноважень або ресурсів для впровадження масштабної інформаційної системи;

4. Комерційний відділ

Роздрібна та оптова торгівля координується вручну, із застосуванням окремих сервісів для клієнтів та таблиць для фіксації замовлень. Магазини

надсилають запити у вигляді Google таблиць, а звіти та аналітика виконуються вручну.

Отже, установа хоч і використовує окремі електронні засоби, але єдина інформаційна система, яка б об’єднувала діяльність всіх відділів відсутня. Такий рівень комп’ютаризації не відповідає сучасним вимогам для ефективного управління. Тому проєктування та впровадження централізованої інформаційної системи є необхідним для покращення продуктивності роботи всієї установи.

1.4. Функціональне моделювання та аналіз існуючих бізнес-процесів

1.4.1. Функціональна модель

Комерційний відділ, а саме робітники підвідділу роздрібних продажів регулярно отримують прохання клієнтів щодо адаптування зручного сайту для мобільних пристроїв, або розробку мобільного застосунку. Покупці часто не мають можливості оформити замовлення або переглянути товар у зручному форматі, якщо знаходяться за межами дому.

При відкриванні головної сторінки сайту на ПК користувач може переглянути каталог товарів, які можуть бути відібрані по категоріям та мати скорочену інформацію про товар та його характеристики. На сторінці кожного продукту є функція додавання товару у кошик, а при оформленні замовлення, персональні дані та адресу для доставки потрібно вписувати вручну. Оскільки, на сайті відсутня можливість мати свій особистий профіль, то і, зрозуміло, відсутня система накопичення бонусів.

Модель бізнес-процесів у нотації BPMN рівня AS-IS наведена на рисунку 1.3.

Недоліки бізнес-процесів, виявлені в процесі моделювання:

1. Відсутність мобільного додатку, що знижує доступність магазину для більшого кола користувачів;
2. Складна навігація по сайту, оскільки він не є підлаштованим під мобільний пристрій. Через дрібний шрифт та незрозумілу структуру цілком можлива втрата клієнта;
3. Відсутність особистого кабінету. Це означає відсутність історій замовлень, системи бонусів чи персональних даних клієнта;

4. Відсутність системи лояльності. Що може призвести до втрати клієнта, через надто малу мотивацію повернутися та зробити повторне замовлення;

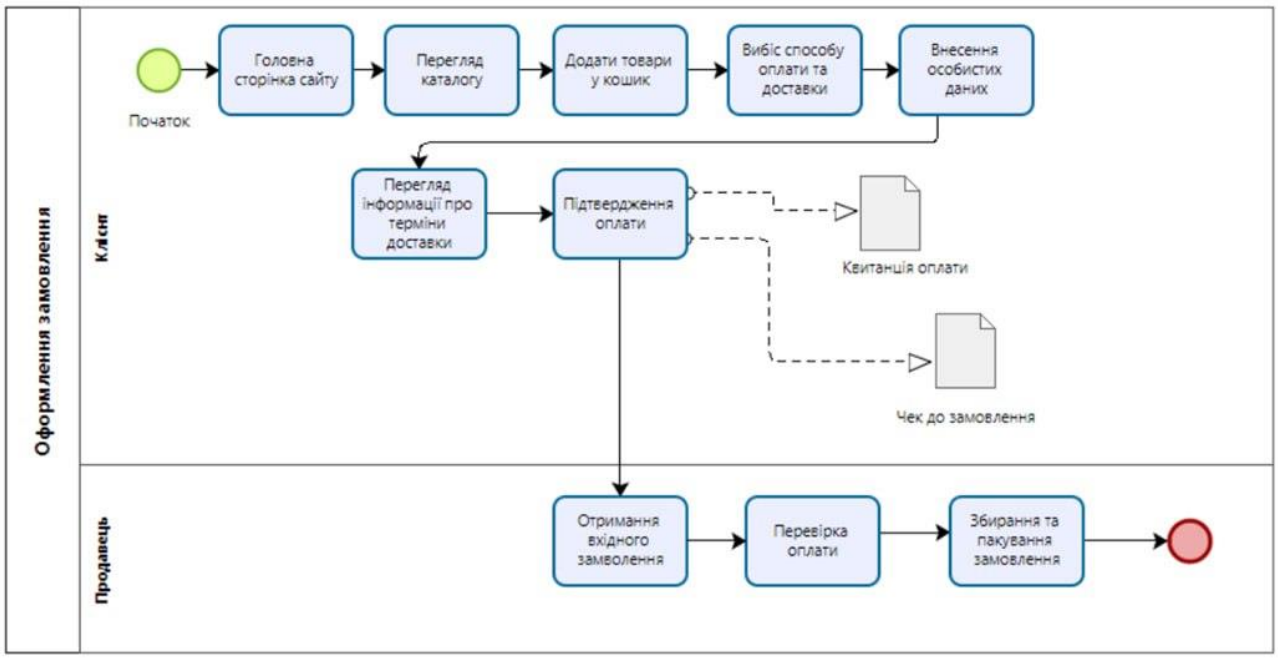

Рисунок 1.3 – Модель бізнес-процесів у нотації BPMN рівня AS-IS

1.4.2. Пропозиція щодо вдосконалення бізнес-процесів

Створення мобільного додатку, щоб задовольнити потреби клієнтів та збільшити прибуток магазину та додання функцій, які майже повністю задовольняють потреби магазину. Створений додаток має список функцій, які покращать ефективність роботи, а саме:

1. Вхід в особистий кабінет, або реєстрація у додатку;
2. Впроваджена система лояльності для покупців;
3. Зручний інтерфейс для мобільних пристроїв;
4. Історія попередніх замовлень;

На рисунку 1.4. наведена модель бізнес-процесів у нотації BPMN рівня TO-BE.

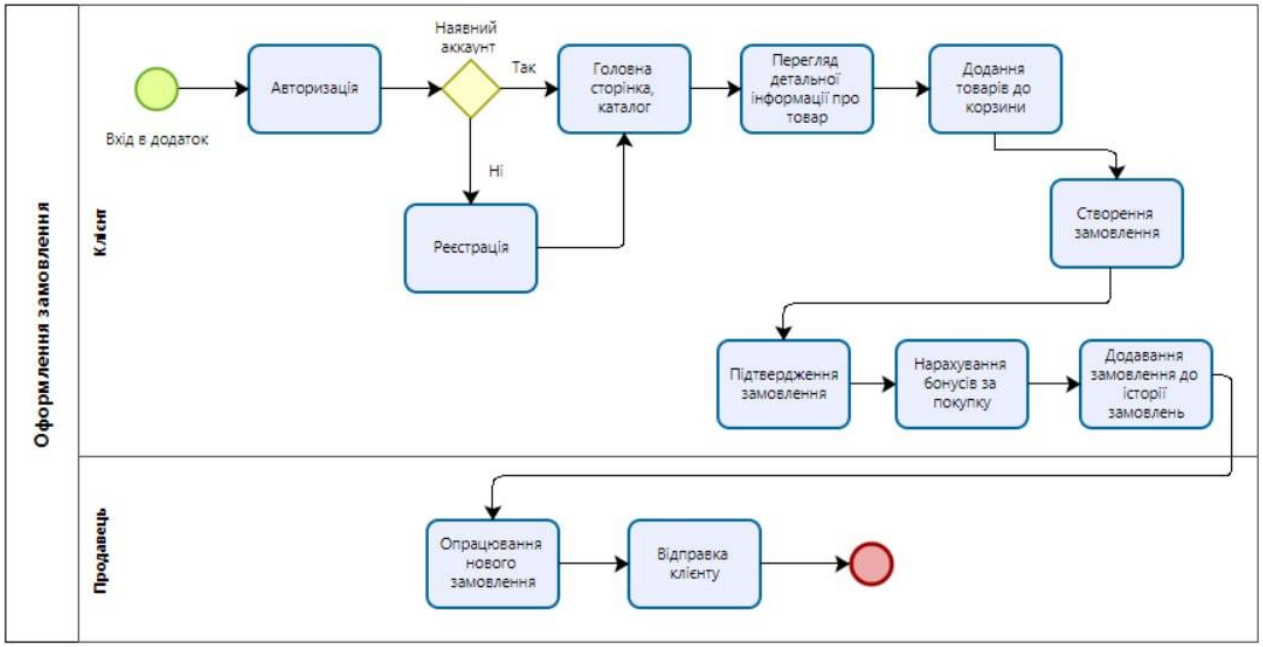

Рисунок 1.4 – Модель бізнес-процесів у нотації BPMN рівня TO-BE

1.5. Огляд існуючих рішень

Для створення ефективного додатку, було прийнято рішення переглянути, оцінити та порівняти інформаційні системи-аналоги інших магазинів.

1.5.1. Онлайн-магазин «AsiaFood»

AsiaFood – невеликий онлайн-магазин товарів з Азії: локшина, соуси, снеки, напої з Кореї, Японії та Таїланду. Хоч мобільного застосунку у них і нема, але сайт магазину зручно підлаштований до користування з телефону.

Головним перевагами сайту є:

- Каталог з фільтрами, де товари поділені не тільки за типом, а і за країнами виробниками;
- Оформлення замовлення без обов’язкової реєстрації;
- Наявна мобільна адаптація сайту.

1.5.2. Магазин «J-food»

J-food – магазин, що спеціалізується на японській та корейській кухні, імпортованих продуктах, включаючи напої та приправи для готування. Мобільний додаток таж відсутній, але сайт чудово підлаштований під мобільного користувача.

Переваги сайту магазину:

- Створені окремі категорії «Новинки», «Акції», «Хіти продажів» для клієнтів, які не впевнені в своєму виборі, або, навпаки, полюють на наявність новинок;
- Блог з рецептами, який показує не тільки спосіб приготування продуктів, а і має поради щодо кращого смаку;
- Функція зворотного зв'язку – користувач може залишити номер і менеджер зв'яжеться з ним.

1.5.3. Магазин «Korea Mart»

Korea Mart – магазин товарів не тільки з Кореї, Японії і Китаю, але й має унікальну та цікаву продукцію з Тайваню та В'єтнаму, що робить його досить унікальним магазином. Сайт магазину є зручним для користування і з ПК і з мобільного пристрою.

Переваги сайту:

- Наявна система бонусів (балів) які накопичуються за покупки;
- Наявність особистого кабінету з історією замовлень;
- Можливість обрати мову сайту;
- Зручна мобільна адаптація.

Проаналізувавши складові сайтів у магазинів аналогів, стало зрозуміло, що більшість з них вже адаптувалися під користувачів телефонів. Але у даних випадках магазини мають лише мобільну версію сайту, а це означає, що розробка окремого додатку для магазинів «Смак Кореї» є актуальною та навіть конкурентно спроможним рішенням.

1.5.4. Порівняння систем-аналогів

Нижче у таблиці 1.4 наведено порівняння функціоналу існуючих рішень для популярних магазинів корейської продукції.

Таблиця 1.4. Порівняння систем-аналогів магазинів корейської продукції

Наявні функції	**AsiaFood**	**J-food**	**Korea Mart**
Каталог товарів	Є	Є	Є
Пошук за назвою	Є	Є	Є
Фільтр за категоріями	Є	Є	Є
Розділ «Новинки» та «Акція	Немає	Є	Є
Рецепти приготування	Немає	Є	Немає
Швидке замовлення	Немає	Є	Немає
Особистий кабінет	Немає	Немає	Є
Історія замовлень	Немає	Немає	Є
Система бонусів	Немає	Немає	Є

Аналіз поточного стану бізнес-процесів і порівняння з конкурентами свідчить про нагальну потребу у власній розробці мобільного застосунку для магазину «Смак Кореї». Відсутність адаптації сайту до мобільних пристроїв, особистого кабінету, системи лояльності та інших зручних функцій значно знижує зручність користування сервісом та утримання клієнтів. У той час як конкуренти вже мають мобільно-орієнтовані веб-рішення, створення повноцінного додатку дозволить не лише покращити клієнтський досвід, а й забезпечити конкурентну перевагу, підвищити рівень обслуговування, лояльність покупців і, відповідно, збільшити прибуток.

1.6. Техніко-економічне обґрунтування впровадження програмного забезпечення

Для ефективної розробки та впровадження мобільного додатку у мережу магазинів «Смак Кореї» було виконано розрахунки витрат на ключові показники реалізації проєкту.

Основні етапи проекту:

1. Планування

- Трудомісткість (ТМ): 50 люд.-днів
- Чисельність працюючих (ЧП): 2 особи

- Тривалість (ТР): 25 днів
- Витрати:
    - Зарплата персоналу: 36 000 грн.
    - Консультації з аналітиками: 24 000 грн.
    - Разом: 60 000 грн.

2. Проєктування

- Трудомісткість (ТМ): 90 люд.-днів
- Чисельність працюючих (ЧП): 3 особи
- Тривалість (ТР): 30 днів
- Витрати:
    - Зарплата персоналу: 72 000 грн.
    - Платформи для дизайну та прототипування: 32 000 грн.
    - Разом: 104 000 грн.

3. Розробка та налаштування системи

- Трудомісткість (ТМ): 320 люд.-днів
- Чисельність працюючих (ЧП): 4 особи
- Тривалість (ТР): 80 днів
- Витрати:
    - Зарплата розробників: 480 000 грн.
    - Інструменти для розробки: 160 000 грн.
    - Разом: 640 000 грн.

4. Тестування

- Трудомісткість (ТМ): 120 люд.-днів
- Чисельність працюючих (ЧП): 3 особи
- Тривалість (ТР): 40 днів
- Витрати:
    - Зарплата тестувальників: 200 000 грн.
    - Ліцензії для тестування: 40 000 грн.
    - Разом: 240 000 грн.

5. Налаштування

- Трудомісткість (ТМ): 40 люд.-днів
- Чисельність працюючих (ЧП): 2 особи
- Тривалість (ТР): 20 днів
- Витрати:
    - Зарплата системних адміністраторів: 80 000 грн.
    - Сервери та інфраструктура: 80 000 грн.
    - Разом: 160 000 грн.

6. Навчання персоналу

- Трудомісткість (ТМ): 30 люд.-днів
- Чисельність працюючих (ЧП): 2 особи
- Тривалість (ТР): 15 днів
- Витрати:
    - Тренінги для персоналу: 80 000 грн.
    - Разом: 80 000 грн.

7. Підтримка та запуск

- Трудомісткість (ТМ): 750 люд.-днів
- Чисельність працюючих (ЧП): 4 особи
- Тривалість (ТР): 185 днів
- Витрати:
    - Підтримка системи: 60 000 грн.
    - Оновлення платформ: 20 000 грн.
    - Разом: 80 000 грн.

Загальні витрати:

60 000 грн. (планування) + 104 000 грн. (проєктування) + 640 000 грн. (розробка та налаштування системи) + 240 000 грн. (тестування) + 160 000 грн. (налаштування) + 80 000 грн. (навчання персоналу) + 80 000 грн. (підтримка та запуск) = 1 340 000 грн.

Очікуваний економічний ефект:

1. Зниження операційних витрат

Завдяки автоматизації управління товарами, обробки замовлень та підтримки клієнтів очікується зниження витрат на обслуговування на 18%. Це дозволить зекономити 280 000 грн. на рік.

2. Збільшення обсягів продажу

Автоматизована система дозволить обробляти до 30% більше замовлень завдяки покращеній роботі сайту та інтеграції з платіжними системами. Це призведе до збільшення доходів на 360 000 грн. на рік.

3. Скорочення часу на обробку замовлень

Оптимізація процесів дозволить скоротити час виконання замовлень на 25%, що принесе додаткову економію в розмірі 120 000 грн. на рік.

Загальний економічний ефект:

- 280 000 грн. (зниження витрат)
- 360 000 грн. (збільшення доходів)
- 120 000 грн. (економія на часі)

Загальна економія: 760 000 грн. на рік.

Окупність проекту

Термін окупності проекту розраховується за формулою:

$$\text{Термін окупності} = \frac{\text{Витрати на проєкт}}{\text{Річний економічний ефект}} = \frac{1{,}340{,}000}{760{,}000} \approx 1{,}76 \text{ року}$$

Отже, термін окупності складає приблизно 1,76 року.

Отже, проєкт окупився і відси можна зробити висновок, що створення мобільного додатку є не тільки вигідним у плані терміну окупності, а і обов'язковим для відповідності сучасним потребам магазинів та задоволенні клієнтів.

1.7. Обґрунтування доцільності проєктування й розроблення мобільного додатку для мережі магазинів «Смак Кореї»

Проєктування мобільного додатку – це безумовно доцільне рішення, яке допомагає магазину не тільки відповідати сучасним вимогам, а і задовольняє бажання покупців та робить його конкуренто спроможним. Оскільки все більше

людей цінують свій час, то переваги швидкого використання додатку є очевидними для залучення ще більшої кількості клієнтів.

Мобільний застосунок дозволяє покупцю створити особистий профіль, переглянути історію своїх замовлень, накопичувати, і згодом списувати, бонуси, що значно підвищує комфорт користування та спонукає покупця до повторних покупок. Додаткова перевага додатку – це можливість швидко та ефективно сповіщати про наявність акцій чи новинок у магазині, що є фінансово вигідним рішенням для обох сторін.

Крім того, власний створений додаток дозволяє мережі магазинів виділитися серед конкурентів, особливо серед не дуже великих конкурентів, подібних типів, де ще не всі мають такі технологічні рішення. Це створює образ сучасного та прогресивного магазину, який прислухується до прохань клієнтів.

РОЗДІЛ 2. ТЕХНІЧНЕ ЗАВДАННЯ НА ПРОЄКТУВАННЯ

2.1. Загальні положення

2.1.1. Найменування системи

«Розроблення клієнтського мобільного додатку мережі магазинів Смак Кореї».

2.1.2. Результати робіт

Результати робіт із створення системи оформлюються згідно з вимогами ДСТУ на відповідні етапи розроблення. Порядок оформлення і передачі результатів визначається змістом і календарним планом виконання робіт.

2.1.3. Зміни та уточнення

У випадку необхідності на наступних стадіях зі створення системи окремі положення можуть уточнюватися і розвиватися за погодженням із замовником.

2.2. Призначення і цілі створення системи

2.2.1. Призначення системи

Система призначення для покращення обслуговування клієнтів мережі магазинів «Смак Кореї» шляхом швидкого та зручного доступу до інформації про актуальну наявність, ціну, опис продуктів. Окрім цього, у додатку реалізовано функції пошуку та фільтрації асортименту, реєстрації та авторизації користувачів, створення замовлень та зберігання інформації про них, накопичування та списання бонусів з замовлень.

2.2.2. Цілі створення системи

Основна ціль створення додатку це зручна та швидка взаємодія клієнтів з асортиментом магазину з метою покращення користувацького досвіду, зменшення витрат часу перегляд асортименту та створення замовлень, що позитивно вплине на загальну роботу всіх бізнес процесів системи.

Цілі створення:

- покращення якості обслуговування клієнтів за рахунок зручного доступу до інформації про асортимент;
- надання можливості створення онлайн замовлень;

- персоналізація досвіду користувача за допомогою системи збереження виконаних замовлень та накопичення бонусів;
- оптимізація взаємодії між покупцем та магазином – зменшення навантаження на фізичні магазини;
- підвищення лояльності клієнтів завдяки зручному, сучасному та інтуїтивно зрозумілому інтерфейсу.

2.3. Характеристика об’єкта автоматизації

2.3.1. Короткі відомості про об’єкт автоматизації

Об’єкт автоматизації це комерційний відділ мережі магазинів, який взаємодіє з клієнтами через мобільний додаток. У зв’язку з активним розвитком мережі Смак Кореї як постачальника корейських товарів та зростаючим попитом серед покупців, виникла необхідність у цифровізації та вдосконаленні обслуговування клієнтів з метою підвищення лояльності та підтримки позитивного іміджу компанії.

Основні функції, які потребують автоматизації:

- перегляд повного асортименту товарів з характеристиками;
- фільтрація за наявними характеристиками (назва, ціна, категорія, гострота);
- доступ до особистого кабінету користувача з історію замовлень та переглядом накопичених бонусів;
- додавання товарів в кошик та створення замовлень.

2.4. Вимоги до системи

2.4.1. Вимоги до системи в цілому

2.4.1.1. Вимоги до структури і функціонування системи

Архітектура системи

Система повинна маю клієнт-серверну архітектуру з використанням таких технологій:

- мобільний додаток на платформі Android (клієнт) – .NET MAUI C#;
- серверна частина – ASP.NET Core Web API;

- база даних – SQL Server з адмініструванням через SQL Server Management Studio (SSMS).

Діагностування функціонування системи

Система надає діагностику помилок та збоїв користувачу за допомогою короткого сповіщення. Без виправлення даної помилки подальша робота з системою не можлива.

Можливості розвитку і модернізації

Система передбачає можливість розширення функціоналу шляхом простого додавання нових функцій та модулів у код без потреби повного переписування. Завдяки використанню кросплатформного фреймворку .NET MAUI, додаток легко масштабується та може бути реалізований не лише для Android, а й для iOS.

Режими функціонування

Функціонування системи забезпечує повне функціонування при наявності інтернету.

2.4.1.2. Вимоги до користувачів системи

Категорії користувачів:

Додаток передбачає наявність двох типів користувачів:

1. Клієнт – може реєструватись та авторизовуватись в додатку, переглядати асортимент товару, здійснювати фільтрацію по характеристикам товарів, додавати товар в кошик, формувати замовлення, переглядати виконані замовлення, переглядати та списувати накопичені бонуси;
2. Розробник – має повний доступ до системи, працює з базою даних продуктів та клієнтів, надає підтримку та оновлює додаток.

2.4.1.3. Вимоги до кваліфікації користувачів:

Для ефективного користуванням додатком користувачі мають:

- мати базові навички роботи з мобільними пристроями, вміти встановлювати додатки з Google Play Market;
- розуміти принцип роботи додатку, включаючи навігацію між розділами, фільтрацію товарів та створення замовлень;

- знати типові ситуації, що можуть спричинити помилки (втрата інтернет-з’єднання, недоступність серверу) та розуміти базові дії для їх вирішення (перезапуск додатку, оновлення сторінки, перевірка підключення до інтернету).

2.4.1.4. Показники призначення

Характеристики якості системи

Система повинна забезпечувати:

1. Швидкість завантаження головної сторінки після входу в аккаунт користувача – не більше 3 секунд;
2. Пошук товару методом введення назви без додаткових кнопок – не більше 1 секунди;
3. Швидкість переходу між вкладками додатку – не більше 1 секунди.

Показники надійності

Основні показники надійності системи:

- ймовірність безвідмовної роботи – не менше 0,94;
- максимальний час простою системи – не більше 4 годин на місяць.

Забезпечення надійності

Для забезпечення надійності необхідно передбачити:

- автоматичне резервне копіювання даних не менше 60 днів.;
- моніторинг працездатності системи в режимі реального часу;
- механізм перевірки цілісності даних при їх передачі.

2.4.1.5. Вимоги до безпеки

Для забезпечення безпеки при експлуатації системи потрібно дотримуватись вимог ДСТУ: ДСТУ 2293-99, ДСТУ ISO 6309:2007, ДСТУ 12.0.230:2008, ДСТУ 7237:2011, ДСТУ 7238:2011, ДСТУ 7239:2011.

2.4.1.6. Вимоги з ергономіки та технічної естетики

Загальні ергономічні і естетичні вимоги до системи повинні відповідати держстандартам ДСТУ 8604:2015, ДСТУ 7298:2013. Інтерфейс ПК додатку повинен відповідати сучасним тенденціям UI/UX дизайну та загальноприйнятим принципам побудови інтерфейсів для настільних комп'ютерів та ноутбуків.

2.4.1.7. Вимоги до експлуатації

Види обслуговування

Види обслуговування системи визначаються у відповідності з ДСТУ EN 13306:2019. Загальні вимоги з експлуатації, технічного обслуговування і ремонту повинні відповідати ДСТУ 3576-97.

2.4.1.8. Вимоги до технічних засобів

Для коректної роботи інформаційної системи рекомендується мати такі необхідні технічна засоби:

- телефон з операційною системою Android не нижче версії 8.0 (Oreo);
- оперативна пам'ять: не менше 3 ГБ;
- вільна пам’ять на пристрої: не менше 300 МБ;
- стабільне підключення до мережі інтернет.

2.4.1.9. Вимоги до захисту інформації від несанкціонованого доступу

Для надійності збереження інформації необхідно застосувати такі засоби захисту як:

- автентифікація користувачів за допомогою логіна та пароля;
- шифрування персональних даних у базі даних;
- розмежування прав доступу до функцій системи.

2.4.1.10. Вимоги щодо збереження інформації при аваріях

Засоби резервного збереження

Засоби резервного збереження даних повинні передбачати:

- щотижневе повне резервне копіювання даних;
- зберігання резервних копій протягом щонайменше 3 місяців;
- автоматичне відновлення даних у разі збоїв системи.

Розміщення резервних копій

Резервні копії слід зберігати на окремих носіях даних, доступ до яких має керівник системи з різних пристроїв для забезпечення максимальної надійності.

2.4.1.11. Вимоги щодо захисту від впливу зовнішніх діянь

Фізичний захист обладання

Доступ до серверів, ПК з резервними копіями, роутерів та іншого критичного обладнання має бути обмежений лише авторизованим персоналом.

Рекомендується використовувати замки, відеоспостереження або контроль доступу з ідентифікацією.

Захист від надзвичайних ситуацій

Приміщення, в якому розміщено серверне обладнання або інфраструктуру збереження даних, повинно бути оснащене автоматичною пожежною сигналізацією, системою пожежогасіння та вентиляцією. Інформаційна система має бути інтегрована з централізованою системою оповіщення про надзвичайні ситуації.

2.4.1.12. Вимоги до патентної частини

Під час розробки системи патентні дослідження не проводяться. Усі використані бібліотеки та компоненти повинні мати відкриту ліцензію, що дозволяє їх комерційне користування.

2.4.1.13. Вимоги щодо стандартизації і уніфікації

У системі кодування інформації необхідно використовувати UTF-8. Дати, час та інші метрики повинні відповідати міжнародним стандартам.

2.4.2. Вимоги до функцій системи

2.4.2.1. Перелік функцій із зазначенням вхідної та вихідної інформації

Перелік функцій з найменуваннями функцій, вхідною та вихідною інформацією подано в таблиці 2.1.

Таблиця 2.1. Перелік функцій, вхідної та вихідної інформації

Найменування функції	**Вхідна інформація**	**Вихідна інформація**
Реєстрація користувача	Логін, пароль, ім’я користувача	Створений обліковий запис із відповідним рівнем доступу, створено код бонусної картки
Автентифікація користувача	Логін та пароль користувача	Ім’я, логін, код бонусної карки користувача, список оформлених замовлень

Найменування функції	**Вхідна інформація**	**Вихідна інформація**
Пошук та фільтрація товару	Назва товару або її частина, обрана категорія, межі цін та гостроти	Список товару відповідно до вхідних фільтрів
Перегляд картки товару	Обраний товар	Назва товару, фото товару, ціна, опис, склад продукту харчування, вага, рівень гостроти
Додавання товару в кошик	Обрані товари, кількість товару в позиції	Загальна сума замовлення, оформлене замовлення, списані або нараховані бонуси
Перегляд списку замовлень	Пошуковий запит (найменування, місце зберігання, виробник, категорія)	Інформація про товар (його місцезнаходження, кількість та термін)
Перегляд списку замовлень	Перехід на вкладку Профіль користувача через інтерфейс	Список оформлених замовлень, номер, дата, сума замовлення, кількість нараховних або списаних бонусів
Перегляд картки оформленого замовлення	Перехід на відповідну картку оформленого замовлення через інтерфейс	Номер замовлення, список позицій в замовленні, сума та кількість в позиції, загальна сума в замовленні

2.4.3. Вимоги до видів забезпечення

2.4.3.1. Вимоги до математичного забезпечення

Система не потребує спеціального математичного забезпечення для виконання своїх функцій — достатньо можливостей, які надають обрані технології.

2.4.3.2. Вимоги до інформаційного забезпечення

Організація інформаційного забезпечення

Інформаційна забезпечення системи має включати дані, необхідні для повноцінного виконання всіх функцій, покладених на систему.

Перелік сутностей:

- користувачі;
- категорії товарів;
- товари;
- замовлення;
- деталі замовлень.

Захист даних

Необхідно передбачити захист даних від втрати у разі аварій або збоїв у енергопостачанні системи шляхом використання резервного копіювання бази даних.

2.4.3.3. Вимоги до лінгвістичного забезпечення

Мови розроблення

Для розроблення програмних засобів повинні використовуватися:

- мова SQL для роботи з базою даних;
- API для підключення бази даних;
- C# для роботи з UI.

Організація діалогу

Взаємодія користувача із системою повинна забезпечуватися за допомогою інтуїтивно зрозумілого інтерфейсу, заснованого на використанні стандартних компонентів мобільних застосунків — таких як кнопки, меню, форми введення тощо.

2.4.3.4. Вимоги до програмного забезпечення

Загальносистемне програмне забезпечення

До загальносистемного забезпечення відноситься:

- Операційна система мобільних пристроїв користувачів Android не нижче версії 8.0 (Oreo);
- Система управління базами даних (СУБД) — реляційна СУБД Microsoft SQL Server.

Загальні вимоги до системного ПЗ

- Ефективне використання апаратних ресурсів мобільних пристроїв та серверної частини;
- Висока продуктивність у роботі мобільного додатку та API;
- Надійність та безпека захисту даних під час обробки;
- Повна відповідність визначеним функціональним вимогам.

Вимоги до технологій розробки

- Автоматизація процесів за допомогою CI/CD-підходу з використанням GitHub Actions або Jenkins;
- Використання бази даних (SQL Server) для зберігання та обробки даних.

Вимоги до СУБД

- Ефективне управління структурованими даними;
- Висока швидкість виконання запитів між клієнтом і сервером;
- Надійне зберігання даних з можливістю резервного копіювання та відновлення.

Програмні засоби взаємодії з користувачем

- Інтуїтивно зрозумілий користувацький інтерфейс;
- Механізм перевірки та контролю введених користувачем даних;
- Адаптивний дизайн, який коректно відображається на екранах різного розміру.

Вимоги до розробки спеціального ПЗ

- Можливість розширення функціоналу додатку та переносу на інші ОС без значних змін в структурі;
- Оптимізація продуктивності для стабільної роботи на пристроях з різною потужністю;
- Використання сучасних шаблонів проєктування для читабельного та підтримуваного коду.

2.4.3.5. Вимоги до технічного забезпечення

Вимоги до технічного обладання та його основні характеристики наведено у таблиці 2.2.

Таблиця 2.2 Вимоги до технічного забезпечення системи

	Основні характеристики
Серверна частина	CPU: не менше 4 ядер, частота не менше 2.5 GHz RAM: не менше 8 GB HDD/SSD: не менше 100 GB (рекомендовано SSD; можливість RAID) Підключення до мережі: 1 Gbps Ethernet ОС: Microsoft Windows 10 або новіша
Клієнтська частина	CPU: не менше 4 ядер, частота не менше 2.0 GHz RAM: не менше 3 GB Внутрішня пам’ять: не менше 32 GB Екран: від 4" Підключення до мережі: мобільний інтернет або WiFi ОС: Android не нижче версії 8.0 (Oreo)

Обмін інформацією

Обладнання має гарантувати передачу даних у масштабах, необхідних для повноцінного виконання інформаційних потреб системи.

2.4.3.6. Вимоги до метрологічного забезпечення

Система не має вимірювальних каналів, вимірювального обладнання і приладів, тому вимоги до цього виду забезпечення не висуваються.

2.4.3.7. Вимоги до організаційного забезпечення

Стандарти організаційного забезпечення

Організаційне забезпечення системи розробляється відповідно до вимог державного стандарту по АСУП.

Кадрове забезпечення

Під час впровадження системи не планується розширення штату підприємства. Для її обслуговування призначається відповідальна особа.

Вимоги до функціонування системи

- керівник наказом визначає перелік співробітників, які отримуються доступ до системи;
- відповідальна особа контролює роботу системи та приймає рішення у випадку аварійних ситуацій;
- організовується навчання персоналу для ефективної роботи з системою;
- забезпечується дотримання норм безпеки та збереження конфіденційної інформації.

2.5. Склад і зміст робіт по створенню системи

2.5.1. Стадії створення системи і терміни виконання робіт

Стадії створення інформаційної системи та чіткі терміни робіт зазначені у таблиці 2.3.

Таблиця 2.3. Найменування робіт при створенні системи

Етапи проекту	Початок	Тривалість	Затримка	Кінець
Аналіз вимог	28.04.2025	2	0	29.04.2025
Проектування	30.04.2024	4	0	04.05.2025
Розробка	05.05.2025	8	+1	14.05.2025
Тестування	15.05.2025	7	+1	22.05.2025
Впровадження	23.05.2025	3	+1	27.05.2025
Навчання персоналу	28.05.2025	2	-1	30.05.2025
Підтримка та супровід	28.05.2025	3	0	30.05.2025

2.5.2. Діаграма Ганта

Проєкт зазнав незначних коригувань, загалом зміщених на 3 дні, що пов’язано зі змінами в тривалості і затримках на етапах «Розробка», «Тестування», «Впровадження» та «Навчання персоналу». Основні затримки відзначені на етапах «Розробка» (+1 день), «Тестування» (+1 день) і «Впровадження» (+1 день), при цьому навчання персоналу пройшло швидше на 1 день. Незважаючи на це, загальний термін завершення проєкту (30.05.2025) залишився у межах прийнятних строків (див. рисунок 2.1).

Рекомендується посилити контроль за процесами розробки та тестування, а також більш ретельно планувати навчання персоналу, щоб уникнути подібних затримок у подальшому та забезпечити своєчасне виконання всіх завдань.

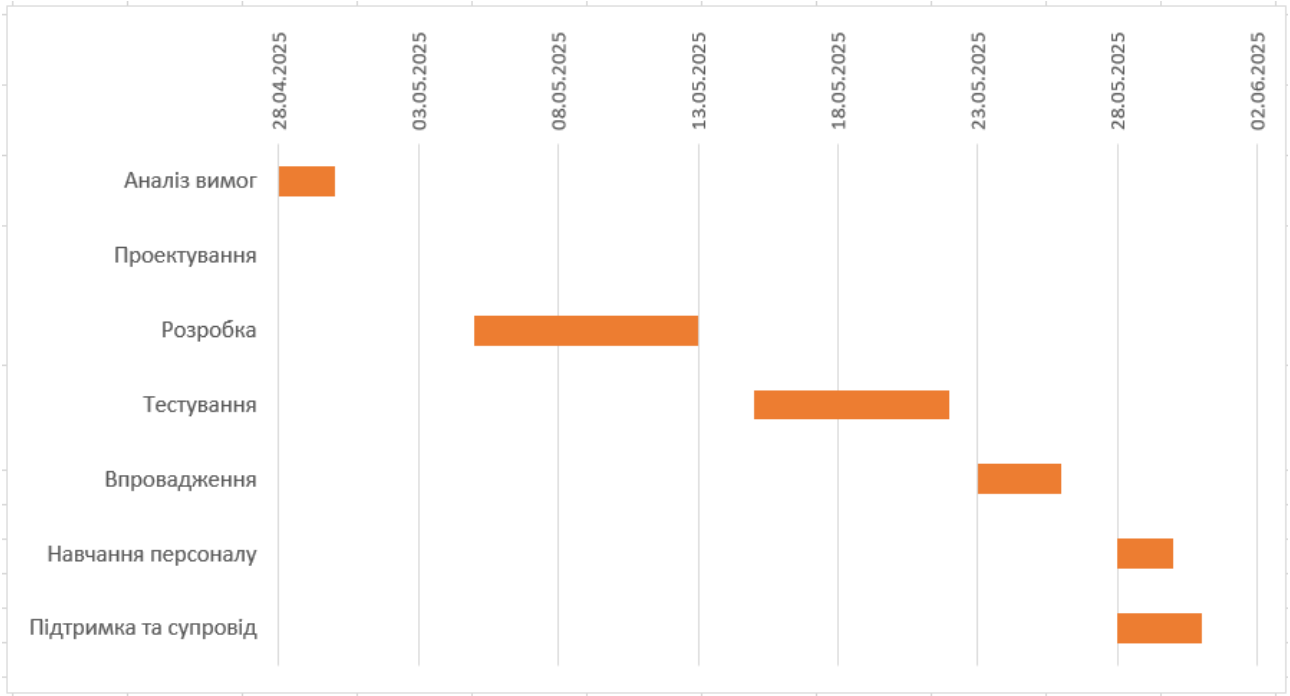

Рисунок 2.1 – Діаграма Ганта

2.6. Порядок контролю і приймання системи

2.6.1. Загальні умови приймання

Система вводиться у діючих магазинах корейської продукції Смак Кореї. При введенні в дію система повинна пройти приймальні випробування згідно з ДСТУ 3974-2000.

2.6.2. Випробування системи

Випробування системи для оцінки її працездатності та прийняття рішення про можливість впровадження в дослідну експлуатацію проводить маркетолог. Його завдання — надати експертну оцінку системи з точки зору зручності та ефективності для кінцевих користувачів, зокрема клієнтів, а також визначити, наскільки функціонал відповідає цілям просування бренду та комунікації з аудиторією.

2.6.3. Дослідна екслуатація

Передача системи в дослідну експлуатацію здійснюється відповідно до технічного завдання та інструкції користувача. За підсумками тестового використання формується список необхідних доопрацювань із зазначенням рекомендованих термінів їх реалізації.

2.6.4. Оформлення результатів

Введення системи в експлуатацію оформлюється актом здачі-прийому, який підписується директором після погодження з відповідальним співробітником (маркетологом) та розробником системи.

2.7. Вимоги до складу і змісту робіт із підготовки до введення системи в дію.

Для успішного впровадження клієнтського мобільного додатку мережі магазинів «Смак Кореї» було проведено такі підготовчі роботи:

- Укомплектування технічних засобів згідно до вимогами технічного забезпечення;
- Організація навчання персоналу роботі з користуванням та адмініструванням мобільного додатку;
- Призначення відповідальних осіб за підтримку та експлуатацію системи;
- Інтеграція із внутрішніми системами обліку та управління товарними запасами;
- Створено тестову групу користувачів для бета-тестування додатку;
- Організовано проведення дослідної експлуатації з подальшим впровадженням на основі аналізу отриманих даних.

2.8. Вимоги до документації

2.8.1. Перелік документації

На систему розробляється комплекс документації у складі:

1. Технічне завдання на розробку мобільного додатку;
2. Технічний проект, що включає:
    - пояснювальну записку;
    - опис архітектури системи;
    - модель даних;
    - керівництво користувачів.
3. Програма та методика випробувань;
4. Акт приймально-здавальних випробувань;
5. Експлуатаційна документація:
    - інструкція з установки інформаційної системи;
    - інструкція з налаштування.

2.8.2. Вимоги до оформлення документації

Документація на систему розробляється у відповідності з вимогами Державних стандартів:

- ДСТУ 3008-2015. Документація. Звіти у сфері науки і техніки. Структура та правила оформлювання;
- ДСТУ 3974-2000 Система розроблення та поставлення продукції на виробництво. Правила виконання дослідно-конструкторських робіт;
- ДСТУ ISO/IEC 25051:2016 Інженерія програмного забезпечення. Вимоги до якості готового до використання програмного продукту (RUSP) і інструкції щодо тестування;
- ДСТУ ISO/IEC/IEEE 26511:2016 Системна та програмна інженерія. Вимоги до менеджерів інформації користувача;
- ДСТУ ISO/IEC/IEEE 26512:2016 Системна та програмна інженерія. Вимоги до укладачів інформації користувача.

2.8.3. Порядок внесення змін до документації

Внесення змін до документації здійснюється у відповідності з ДСТУ ГОСТ 2.503:2013 Єдина система конструкторської документації. Правила внесення змін.

2.9. Джерела розробки

2.9.1. Нормативні документи

При розробленні технічного завдання на систему використано наступні документи:

- ДСТУ 3008-2015. Документація. Звіти у сфері науки і техніки. Структура та правила оформлювання;
- ДСТУ 3973–2000 Система розроблення та поставлення продукції на виробництво;
- ДСТУ Б В.2.5–82:2016 Електробезпека в будівлях і спорудах. Вимоги до захисних заходів від ураження електричним струмом;
- ДСТУ ISO/IEC 27001:2015 Інформаційні технології. Методи захисту. Системи управління інформаційною безпекою. Вимоги;
- ДСТУ ISO/IEC TR 29110-5-1-2:2015 Інженерія програмного забезпечення. Профілі життєвого циклу для малих організацій;
- ДСТУ ISO/IEC/IEEE 29119-1:2015 Тестування програмного забезпечення. Частина 1. Поняття і визначення;
- ДСТУ ISO 9241-210:2019 Ергономіка взаємодії людина-система. Частина 210. Людиноцентричне проєктування інтерактивних систем.

РОЗДІЛ 3. РОЗРОБЛЕННЯ ПРОГРАМНОГО ПРОДУКТУ

3.1. Опис та обґрунтування вибору програмно-технічних засобів розроблення програмного продукту

Для розробки клієнтського мобільного додатку мережі магазинів «Смак Кореї» та серверної частини було обрано сучасний і збалансований технологічний стек, що забезпечує доступність, зручність розробки та ефективність вирішення бізнес-завдань.

Засоби розробки інформаційної системи:

1. Visual Studio – основне середовище розробки, яке використовується для створення мобільного додатку на .NET MAUI, розробки API на ASP.NET Core та адміністрування проекту. Visual Studio надає потужний інструментарій для написання, налагодження та тестування коду, а також інтеграцію з системами контролю версій.
2. .NET MAUI (Multi-platform App UI) – використовується як основна платформа для створення додатку під Android. .NET MAUI дозволяє розробляти єдиний інтерфейс та логіку для різних пристроїв, забезпечуючи адаптивність, високу продуктивність та зручність користувача.

Інтерфейс користувача та функціональність:

1. XAML – використовується для гнучкого створення адаптивного інтрефейсу, що забезпечує зручність на різних пристроях, і спрощує розділення дизайну та логіки.
2. Власні елементи керування – розширюють стандартний набір .NET MAUI, покращують вигляд та функціональність додатку, забезпечують унікальний стиль та зручність користування.

Управління даними та взаємодія між сервером та клієнтом:

1. ASP.NET Core Web API – це швидкий, безпечний та масштабований API для обробки запитів. API забезпечує надійний обмін даними, авторизацію та стабільну взаємодію з базою даних.

2. SQL Server Management Studio (SSMS) – це інструмент для адміністрування бази даних, що дозволяє ефективно керувати структурою даних, виконувати функцію запитів, моніторинг та підтримку БД.

Обґрунтування вибору технологій:

1. Visual Studio з C# та .NET MAUI обрано через:
    - ефективне та потужне середовище розробки з широкими можливостями для кросплатформних мобільних додатків;
    - підтримка сучасних технологій та інтеграція з екосистемою Microsoft;
    - зручні інструменти для розробки UI на основі XAML.
2. ASP.NET Core WEB Api обрано через:
    - ефективність і масштабованість для обробки запитів;
    - вбудовані засоби безпеки та авторизації;
    - стабільну та надійну взаємодію з базою даних.
3. SQL Server обрано через:
    - високу продуктивність при обробці великих обсягів даних;
    - надійність, захист даних і підтримку резервного копіювання;
    - можливість гнучного налаштування інформаційної системи.
4. SQL Server Managmant Studio (SSMS) обрано через:
    - зручний та інтуїтивно зрозумілий інтерфейс для адміністрування бази даних;
    - ефективні інструменти для створення, оптимізації та моніторингу SQL-запитів;
    - доступність та стабільна підтримка.

3.2. Проектування та створення бази даних

У процесі розробки мобільного клієнтського застосунку для мережі магазинів «Смак Кореї» важливою складовою є створення надійної, структурованої та масштабованої бази даних. Саме база даних забезпечує збереження всіх ключових об'єктів системи — товарів, категорій, замовлень, бонусів, клієнтських профілів тощо. Грамотне проєктування бази даних дозволяє

досягти стабільної роботи системи, спростити обробку запитів, а також забезпечити цілісність і консистентність даних.

Для моделювання структури бази даних було використано спеціалізоване програмне забезпечення **ERwin Data Modeler**, яке дозволило створити логічну модель бази даних з чітким визначенням сутностей, зв’язків між ними та атрибутів. Це дало змогу системно опрацювати всі складові частини даних ще до початку фізичної реалізації.

Після завершення побудови логічної моделі в ERwin Data Modeler було здійснено перехід до фізичної моделі бази даних, яка враховує особливості конкретного середовища виконання — SQL Server. На основі логічної структури було автоматично згенеровано SQL-скрипти для створення таблиць, зв’язків, індексів та обмежень. Отримані скрипти імпортовано до середовища SQL Server Management Studio (SSMS), де виконано фізичне створення бази даних. Такий підхід дозволив мінімізувати помилки, забезпечити відповідність між логічною та фізичною структурою та пришвидшити процес реалізації системи.

Під час побудови структури були враховані специфіка торгової діяльності, особливості мобільної взаємодії користувача та вимоги до швидкої обробки замовлень. Структура даних проєктувалась із використанням сучасних принципів нормалізації, а також з урахуванням можливості подальшого розширення функціоналу.

Логічну модель бази даних та опис основних таблиць та зв’язків між ним наведено в додатку В.

Модель містить таблиці для категорій товарів, клієнтів, замовлень, товарів та деталей замовлень, що дозволяє гнучко керувати асортиментом, оформленням та відстеженням замовлень.

Опис сутностей системи:

1. **Category (Категорії)** – містить інформацію про категорії товарів:
    - categoryId – унікальний ідентифікатор категорії;
    - categoryName – назва категорії;

- categoryImage – зображення категорії (опціонально). Використовується для класифікації продуктів у каталозі.

2. **Customers (Клієнти)** – зберігає дані про користувачів додатку:
    - customerId – унікальний ідентифікатор клієнта;
    - fullName – повне ім’я клієнта;
    - login – логін для входу;
    - password – хешований пароль;
    - bonuses – накопичені бонуси клієнта;
    - cardCode – унікальний код бонусної карти. Використовується для аутентифікації та управління лояльністю клієнтів.
3. **Orders (Замовлення)** – зберігає інформацію про оформлені замовлення:
    - orderId – унікальний ідентифікатор замовлення;
    - customerId – ідентифікатор клієнта (зв’язок із таблицею Customers);
    - orderDate – дата та час створення замовлення;
    - totalAmount – загальна сума для контролю процесу замовлення.
4. **Products (Товари)** – містить деталі товарів у каталозі:
    - productId – унікальний ідентифікатор товару;
    - productName – назва товару;
    - categoryId – ідентифікатор категорії (зв’язок із таблицею Category);
    - price – ціна товару;
    - description – опис продукту (опціонально);
    - weight – вага товару (опціонально);
    - spicyLevel – рівень гостроти (опціонально);
    - quantity – доступна кількість товару;
    - productImage – зображення товару (опціонально).

    Використовується для відображення та управління асортиментом.
5. **OrderDetails (Деталі замовлення)** – зберігає інформацію про складові замовлення:

- orderDetailId – унікальний ідентифікатор запису;
- orderId – ідентифікатор замовлення (зв’язок із таблицею Orders);
- productId – ідентифікатор товару (зв’язок із таблицею Products);
- quantity – кількість замовленого товару;
- unitPrice – ціна одиниці товару на момент замовленняю.

Використовується для деталізації замовлення та розрахунку вартості.

На основі побудованої логічної моделі було створено фізичну модель бази даних, яка враховує специфіку обраної системи управління базами даних — Microsoft SQL Server. Структура фізичної моделі оптимізована для ефективного зберігання, швидкої обробки запитів та забезпечення цілісності даних.

Фізичну модель бази даних, що використовується в мобільному застосунку «Смак Кореї» представлено в додатку В.

Після завершення етапу моделювання була реалізована фізична структура бази даних у середовищі **SQL Server Management Studio (SSMS)**, що є офіційним інструментом адміністрування Microsoft SQL Server. Робота виконувалась відповідно до попередньо створеної фізичної моделі, адаптованої для взаємодії з мобільним застосунком.

У процесі реалізації:

- було створено всі необхідні таблиці з урахуванням реляційної структури, визначено типи даних, обмеження, індекси та ключі;
- встановлено зовнішні зв’язки між сутностями (Products, Customers, Orders, тощо) для забезпечення цілісності даних;
- для кожної таблиці передбачено логічне наповнення тестовими записами з метою перевірки функціоналу додатку;
- відбулось тестування на коректність структури бази та перевірка відповідності запитів до потреб застосунку (авторизація, перегляд товарів, обробка замовлень тощо).

Візуальна схема реалізованої структури представлена в Додатку В. Вона демонструє взаємозв’язки між ключовими сутностями та підтверджує

відповідність розробленої бази даних вимогам клієнтського функціоналу мобільного застосунку «Смак Кореї».

3.3. Реалізація функцій системи

3.3.1. Авторизація користувачів

Для забезпечення безпечного та зручного доступу користувачів до додатку реалізовано функцію авторизації у вигляді сторінки LoginPage. Основні можливості:

1. Авторизація користувачів – метод LoginButton_Clicked виконує перевірку введених даних (логін і пароль) та взаємодіє з серверним API:
    - Валідація: перевіряє, чи заповнені поля, чи відповідають логін (5–15 символів) і пароль (8–15 символів) вимогам довжини;
    - Аутентифікація: виконує асинхронний запит до API через ApiService.AuthenticateAsync, який перевіряє правильність введених облікових даних;
    - У разі успішної авторизації завантажує дані користувача через GetCustomerByLoginAsync і зберігає їх у локальних налаштуваннях (Preferences) та у сесії (SessionManager);
    - Після авторизації користувача перенаправляє на головну сторінку додатку (AppShell);
    - Якщо авторизація неуспішна або виникає помилка, показує відповідне повідомлення про помилку.
2. Інтерактивність інтерфейсу – сторінка має зручний дизайн із полями для введення логіну та пароля, відображає повідомлення про помилки у відповідних Label. Також реалізовано посилання на сторінку реєстрації (OnRegisterTapped), що відкриває форму для створення нового акаунту.
3. Збереження стану користувача – після успішної авторизації інформація про користувача (ID, повне ім’я, логін, бонуси, код картки) зберігається у локальних налаштуваннях пристрою та у внутрішній сесії для подальшого використання в додатку.

4. Архітектура і технології – взаємодія з сервером здійснюється через сервіс ApiService, що інкапсулює HTTP-запити, забезпечуючи відокремлення логіки UI від бізнес-логіки і мережевих операцій.

   Код реалізації авторизації користувача представлено в додатку Д.

3.3.2. Реєстрація користувачів

Для реалізації функції створення нового користувача в додатку створено сторінку реєстрації RegisterPage, що забезпечує такі можливості:

1. Реєстрація користувача – метод RegisterButton_Clicked обробляє натискання кнопки реєстрації та виконує послідовну перевірку введених даних:
   - Валідація: перевіряє, чи заповнені поля повного імені, логіну, пароля та підтвердження пароля;
   - Перевіряє довжину повного імені (2–25 символів), логіну (5–15 символів) та пароля (8–15 символів);
   - Порівнює пароль і підтвердження пароля на співпадіння;
   - Виконує асинхронну перевірку унікальності логіну через ApiService.CheckLoginUniqueAsync;
   - Генерує унікальний код бонусної картки методом GenerateCardCode, перевіряючи його унікальність на сервері;
   - Якщо всі перевірки проходять успішно, викликає реєстрацію нового користувача через ApiService.RegisterAsync;
   - У разі успішної реєстрації повертає користувача на попередню сторінку (зазвичай сторінку логіну).
2. Інтерфейс користувача – сторінка має інтуїтивний дизайн з полями введення та підписами помилок, які з’являються при невірних даних. Також реалізовано посилання на сторінку входу, що дозволяє швидко перейти до авторизації (OnLoginTapped).
3. Генерація бонусної картки – для кожного нового користувача генерується унікальний 6-значний код бонусної картки, що перевіряється на унікальність на сервері для запобігання дублікатів.

4. Обробка помилок – у разі помилок (зайнятий логін, невдалий запит до сервера, неправильні дані) користувач отримує відповідне повідомлення, яке допомагає швидко виправити помилки.

Код реалізації реєстрації користувача представлено в додатку Д.

3.3.3. Перегляд та фільтрація товарів

Для реалізації сторінок перегляду товарів та фільтрації в додатку створено AllProductsPage та FiltersPage з такими можливостями:

1. Сторінка AllProductsPage містить такі елементи:
    - Список товарів – товари відображаються в сітці (CollectionView) з двома колонками. Кожен елемент містить зображення, назву, категорію, ціну та кнопку для додавання до кошика;
    - Перехід до деталей: тап на товар відкриває ProductDetailPage з карткою обраного товару;
    - Додавання до кошика: кнопка "+" (OnAddToCartTapped) додає товар у кошик через CartService, перевіряє кількість на складі та показує сповіщення;
    - Пошук – поле пошуку (OnSearchTextChanged) фільтрує товари за введеною назвою або її частиною.
2. Фільтрація – метод FiltersPage фільтрує товари за:
    - Назвою (пошук);
    - Категоріями через CheckBox;
    - Діапазоном цін (MinPrice, MaxPrice);
    - Рівнем гостроти (MinSpicyLevel, MaxSpicyLevel).
3. Завантаження даних – LoadData асинхронно отримує товари та категорії через ApiService. У разі помилки показується сповіщення.
4. Навігація:
    - Кнопка "Назад" (OnBackTapped) повертає на попередню сторінку;
    - Кнопка фільтрів (OnFilterMenuTapped) відкриває FiltersPage;
    - Панель із пошуком, кнопками "Назад" і фільтрів.

5. Картки товарів із зображеннями, ціною, описом продукту та кнопкою додавання в кошик.
6. Обробка помилок – помилки мережі обробляються через DisplayAlert із логуванням.

Код реалізації списку та фільтрації асортименту користувача представлено в додатку Д.

3.3.4. Кошик і замовлення товарів

Для реалізації сторінки кошика в додатку створено CartPage, яка забезпечує такі можливості:

1. Відображення кошика – товари в кошику показуються у списку (CollectionView) із зображенням, назвою, ціною, сумою та кількістю. Користувач може:
    - Видалити товар: кнопка "×" (OnDeleteTapped) видаляє товар із кошика;
    - Змінити кількість: кнопки "+" і "−" (OnIncrementTapped, OnDecrementTapped) змінюють кількість товару, перевіряючи залишки на складі через ApiService.
2. Оформлення замовлення – кнопка "Замовити" (OnPlaceOrderClicked) виконує:
    - Перевірку, чи кошик не порожній;
    - Валідацію введених бонусів: чи є число, чи не перевищує доступні бонуси чи суму замовлення;
    - Перевірку залишків товарів на складі через ApiService.
    - Створення замовлення через ApiService.CreateOrderAsync та додавання деталей (OrderDetail);
    - Оновлення залишків товарів і бонусів клієнта (списання або нарахування 3% від суми);
    - Очищення кошика після успішного замовлення та показ сповіщення з номером замовлення та бонусами.
3. Обробка помилок та показ сповіщень (DisplayAlert) при:

- Порожньому кошику;
- Неправильних бонусах;
- Відсутності авторизації;
- Недостатніх залишках товарів;
- Помилках мережі чи сервера;
- Логування помилок і дій у консоль для дебагінгу.

Код реалізації додавання товарів в кошик та створення замовлення представлено в додатку Д.

3.3.5. Профіль користувача

Для реалізації сторінки профілю користувача в додатку створено ProfilePage, яка забезпечує наступні можливості:

1. Відображення профілю:
   - Сторінка показує персональні дані користувача (вітання з ім’ям, логін, код бонусної картки, кількість бонусів і замовлень);
   - Дані завантажуються асинхронно через LoadUserDataAsync з ApiService за CustomerId із SessionManager або Preferences;
   - Інтерфейс включає кнопку "Вийти" для завершення сеансу.
2. Список замовлень:
   - Відображається через CollectionView із картками, що містять номер замовлення, дату, суму (формат "₴X.XX") та інформацію про бонуси (нараховано чи списано);
   - Тап на замовлення (OnOrderTapped) відкриває OrderDetailsPage із деталями.
3. Вихід із системи – кнопка "Вийти" (OnLogoutClicked) запитує підтвердження, очищає SessionManager і Preferences, перенаправляючи на LoginPage.
4. Обробка помилок і логування:
   - Помилки (відсутність авторизації, проблеми з мережею, відсутність даних) обробляються через DisplayAlert із повідомленнями для користувача;

- Детальне логування через System.Diagnostics.Debug.WriteLine (наприклад, ID клієнта, кількість замовлень, помилки) полегшує дебагінг.

Код реалізації профілю користувача та перегляду створених замволень представлено в додатку Д.

3.3.6. Вкладка «Бонуси»

Для реалізації сторінки управління бонусами в додатку створено BonusesPage, яка забезпечує наступні можливості:

1. Відображення бонусної інформації:
   - Сторінка показує код бонусної картки та кількість бонусів;
   - Дані завантажуються асинхронно через LoadUserDataAsync з ApiService за CustomerId із SessionManager або Preferences.
2. Створення QR-коду:
   - Інтерфейс включає QR-код, згенерований через UpdateQrCode на основі коду картки;
   - Дає можливість сканувати бонусну карту в фізичному магазині для нарахування бонусів
3. Обробка помилок і логування:
   - Помилки (відсутність авторизації, проблеми з мережею, відсутність даних) обробляються через DisplayAlert;
   - Логування через System.Diagnostics.Debug.WriteLine (наприклад, ID клієнта, код картки, помилки) полегшує дебагінг;
   - Якщо код картки відсутній, QR-код не генерується, що запобігає помилкам.

Код реалізації сторінки бонусів та генерації QR-коду представлено в додатку Д.

3.4. Інструкція користувача

Після запуску додатку з’являється сторінка входу (рисунок 3.1), на якій є поля вводу логіну та паролю для авторизації вже існуючих користувачів. При

введенні перевіряється правильність логіну та паролю, при неправильному введенні з’являється відповідне повідомлення.

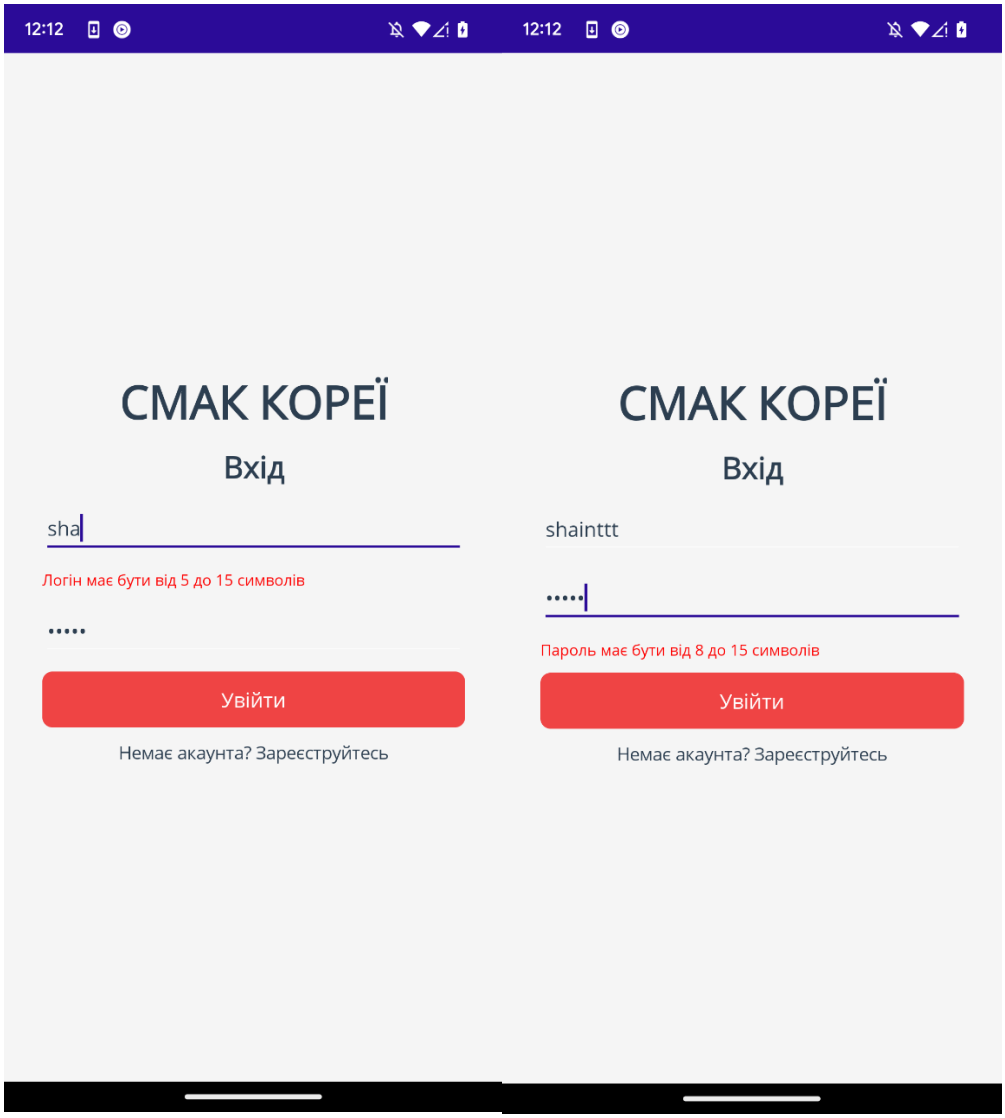

Рисунок 3.1 – Сторінка авторизації з перевіркою вхідних даних

Після натиснення кнопки «Немає акаунта? Зареєструйтесь» відбувається перехід на сторінку реєстрації (рисунок 3.2), де є поля для вводу нового імені, логіну, паролю та повтору паролю. Після введення відбувається перевірка на те, чи заповнені всі поля та на правильність введених даних. При недостатній або надлишковій кількості символів, неспівпадінні паролів або спроби зареєструвати вже існуючий логін під кожним полем з’являється відповідне повідомлення про помилку.

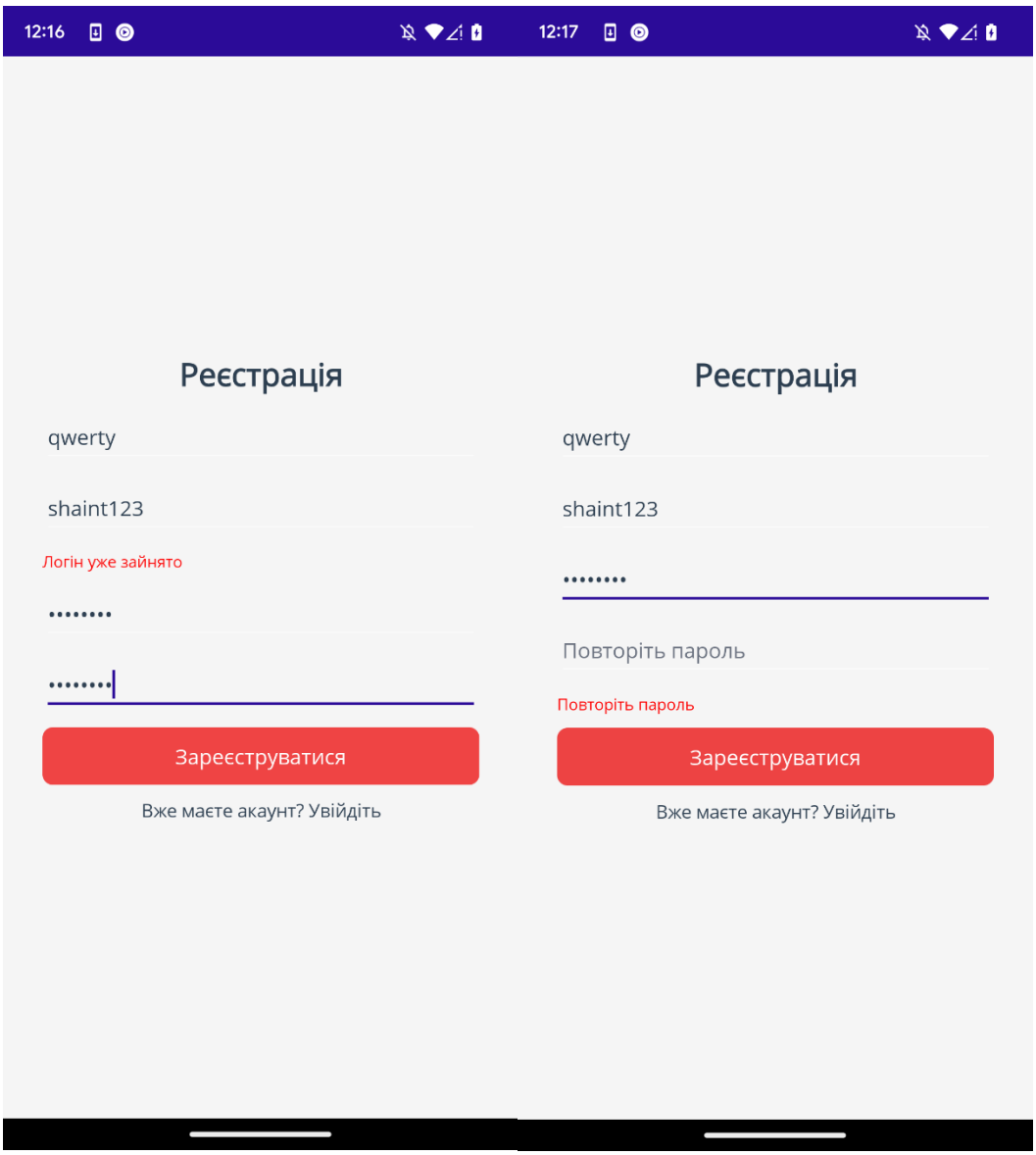

Рисунок 3.2 – Сторінка реєстрації, з перевіркою вхідних даних

Успішна реєстрації переводить додаток назад на сторінку входу. Після успішної авторизації з’являється головна сторінка додатку, та панель внизу екрану, яка дає можливість переключатися між основними сторінками програми (головна, бонуси, кошик, профіль). На головній сторінці (рисунок 3.3) розміщена кнопка всі товари, динамічний список категорій та розділ популярні товари, який містить по одному товару з кожної категорії для демонстрації.

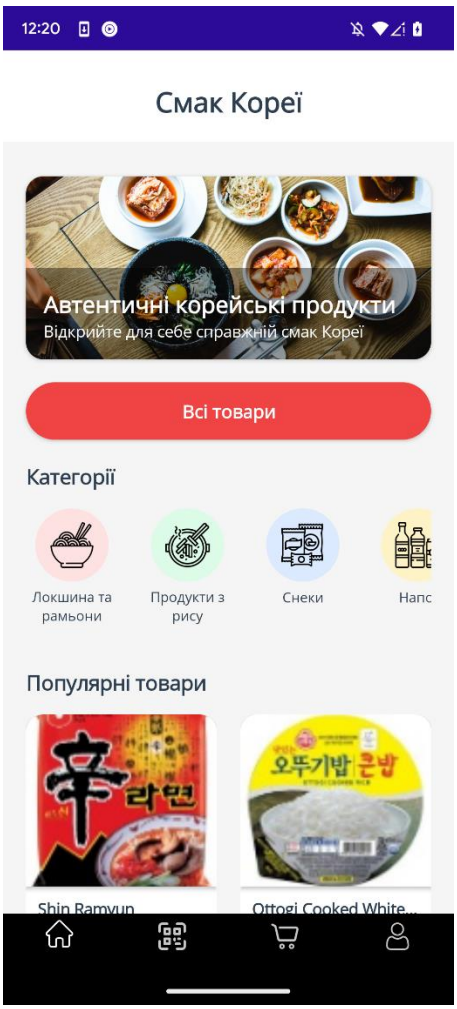

Рисунок 3.3 – Головна сторінка додатку

При переході на сторінку «Всі товари» (рисунок 3.4) відображається поле пошуку, кнопка переходу до фільтрації та список всіх товарів, з фото, ціною та кнопкою додавання в кошик. При введенні в поле пошуку назви товару, або її частини список товарів паралельно фільтрується по введеному значенню.

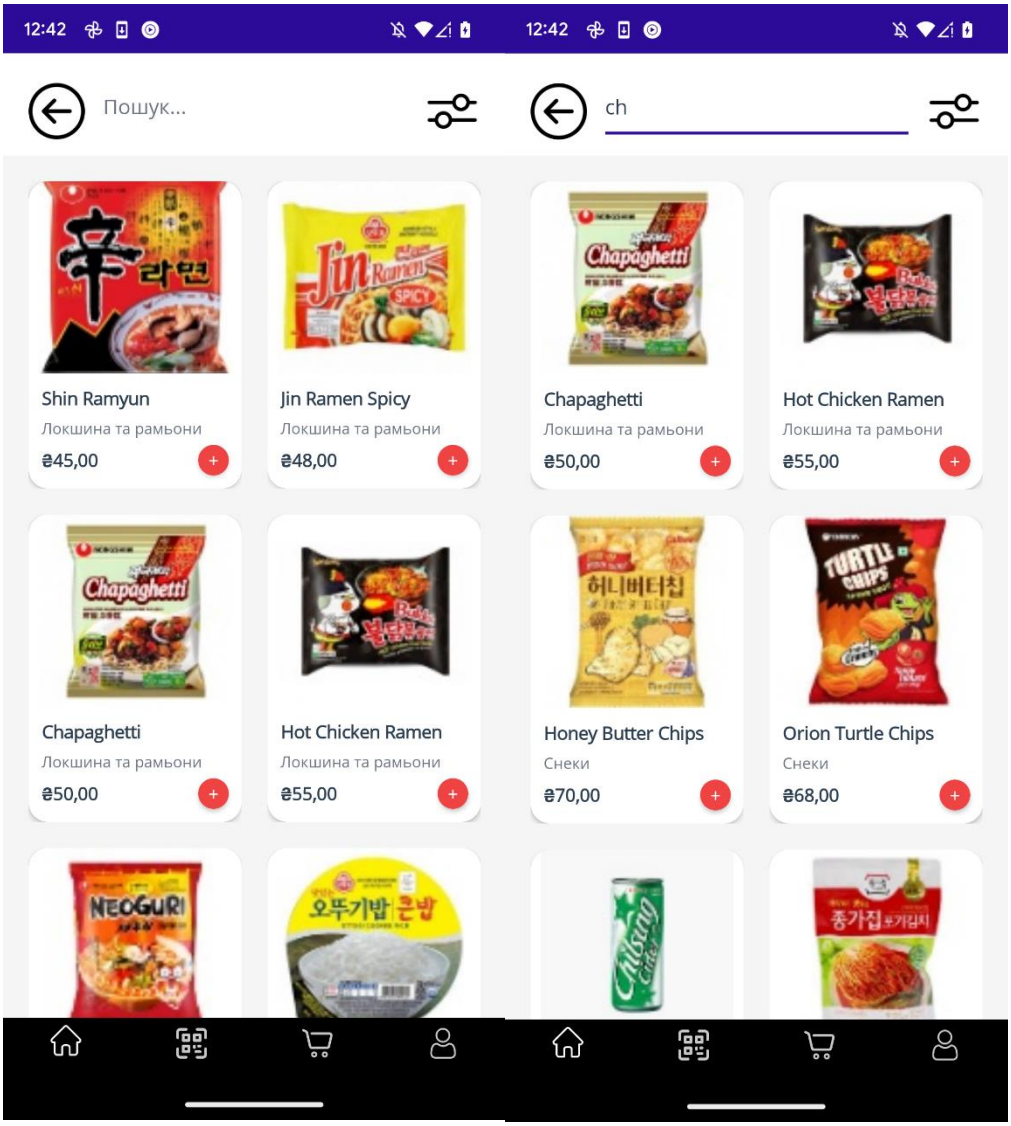

Рисунок 3.4 – Сторінка асортименту з пошуком по назві

Перехід на сторінку фільтрації дає можливість обрати довільну кількість категорій, ціну та гостроту від та до певного значення для фільтрації. При тапі на будь який товар відкривається його картка (рисунок 3.5), з повною інформацією про позицію (фото, назва, ціна, опис, вага, гострота (якщо вона більше 0)), кнопки – та + для регулювання кількості доданих в кошик товарів, та кнопка яка безспосередньо додає обрану кількість товарів до кошик. Натискання на кнопку + або «Додати в кошик» перевіряє, чи достатньо товарів на складі, і при недостачі показує відповідне повідомлення.

Рисунок 3.5 – Картка товару

На сторінці «Кошик» (рисунок 3.6) відображаються позицію, додані до цього через список товарів або картку товару. Кнопки – та + регулюють кількість товару в кожній позиції, кнопка «х» видаляє позицію, сума замовлення при цьому динамічно оновлюється. Під сумою замовлення знаходиться поле з інформацією про доступні для списання бонуси. Натискання на кнопку «Замовити» спочатку перевіряє чи доступні всі товари, чи введені для списання бонуси не перевищують доступні або загальну суму замовлення, після цього формується замовлення. Якщо бонуси не списується, то вони нараховуються на бонусну картку користувача в розмірі 3% від суми замовлення округлені в більшу сторону. Якщо бонуси списуються, то нарахування не відбувається, але

сума сплати за замовлення зменшується на кількість списаних бонусів. Після успішного замовлення з’являється повідомлення з номером замовлення, загальною сумою та нарахованими або списаними бонусами.

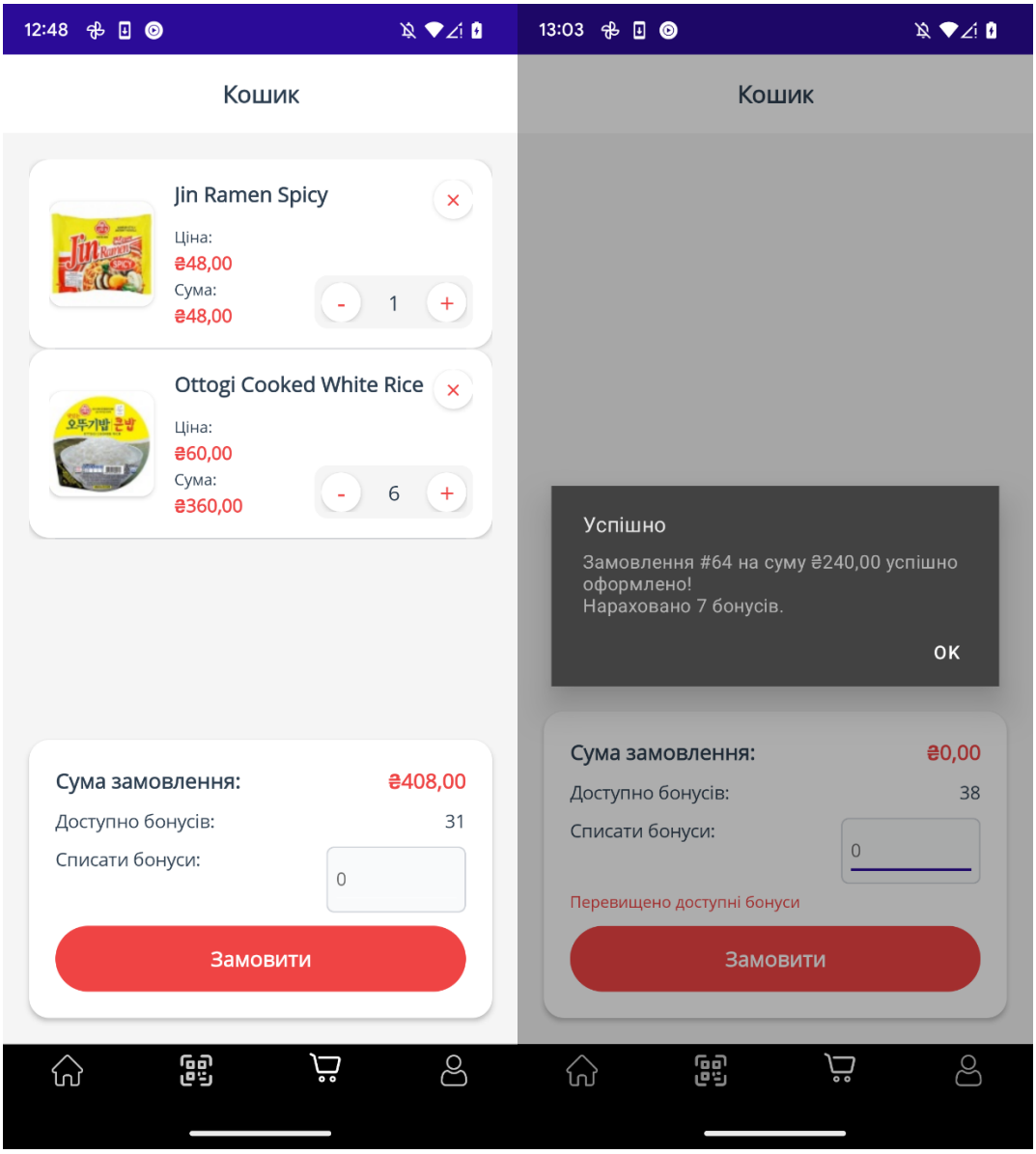

Рисунок 3.6 – Кошик з доданими позиціями та оформлене замовлення

На сторінці «Профіль» (рисунок 3.7) відображається картка профілю, з іменем, логіном та номером бонусної картки користувача, підраховується загальна кількість оформлених замовлень та поточна кількість бонусів. Кнопка «Вийти» завершує сесію поточного користувача, видаляє його дані для входу, та переносить на сторінку авторизації. Під карткою користувача розташовний динамічний список оформлених замовлень, де на кожній позиції відображається

номер замовлення, дата створення, сума та кількість нарахованих або списаних бонусів. Список сортується за датою, останні створені замовлення відображаються зверху.

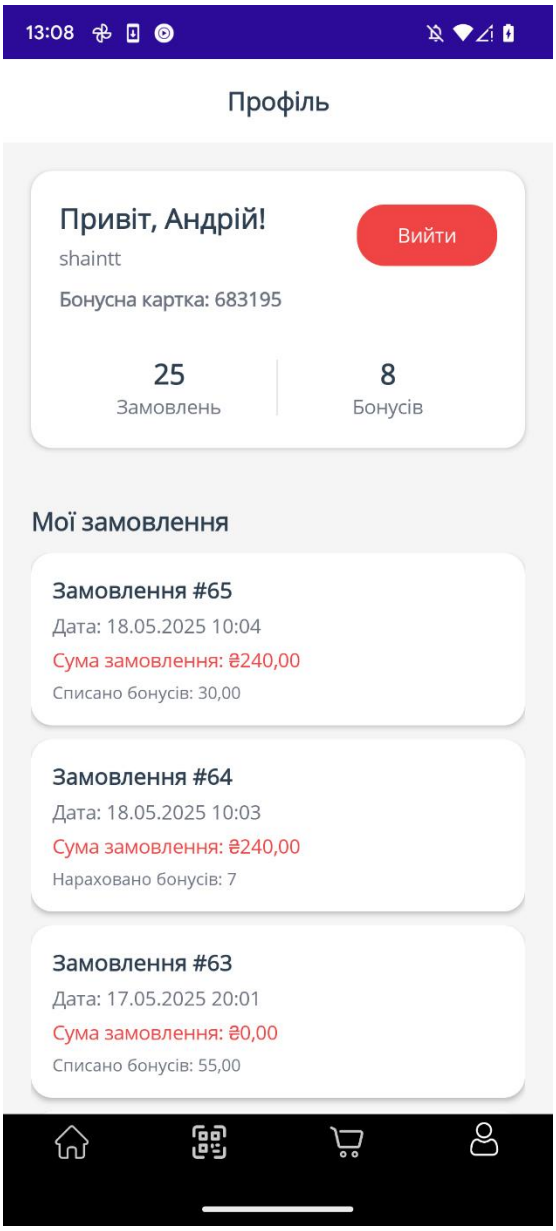

Рисунок 3.7 – Сторінка профілю користувача зі списком оформлених замовлень

При натисканні на замовлення, відкривається сторінка з більш детальною інформацією (рисунок 3.8), де є список товару оформленого в цьому замовленні.

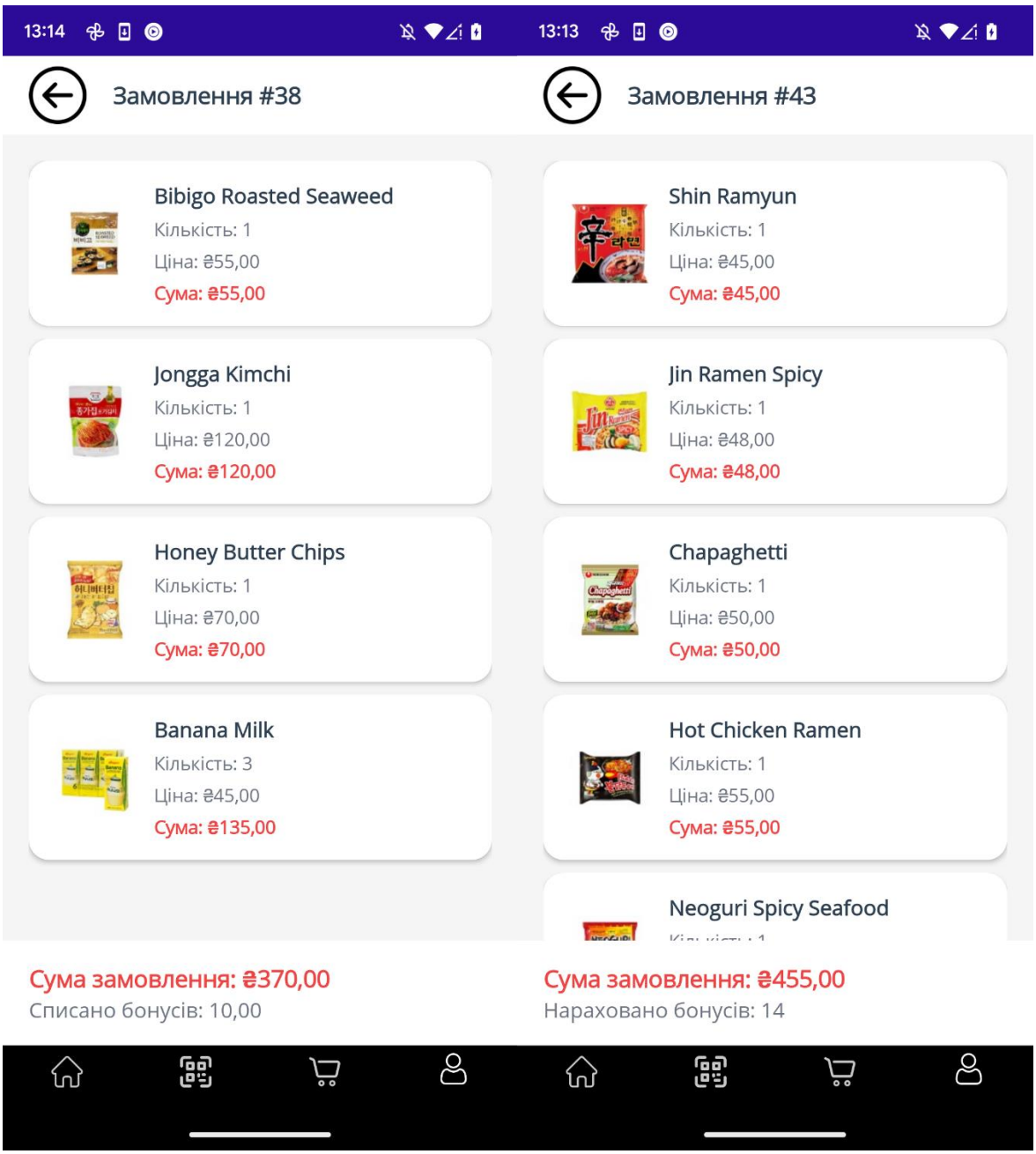

Рисунок 3.8 – Перегляд деталей замовлення

На сторінці «Ваша бонусна картка» (рисунок 3.9) відображається код картки, кількість поточних бонусів, а також по коду картки формується QR-код, який дозволяє нараховувати та списувати бонуси не тільки при замовленнях в додатку, а і в фізичних магазинах «Смак Кореї».

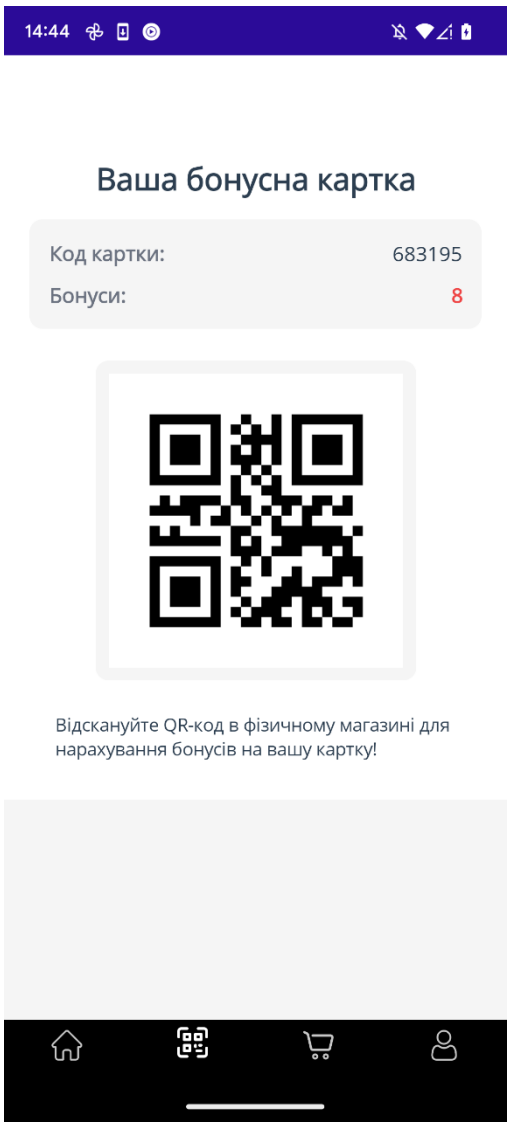

Рисунок 3.9 – Сторінка бонусної картки зі згенерованим QR-кодом

3.5. Тестування програмного продукту

3.5.1. Таблиця результатів тестування

В таблиці 3.1. наведено результати конкретних тесті, які було проведено для перевірки функцій створеної інформаційної системи.

Таблиця 3.1. Результати тестування функцій додатку

№	Пріоритет	Перевірка	Очікуваний результат	Windows 10	Windows 8
1	високий	Вхід користувача (Login)	Авторизація при коректних даних, повідомлення про помилки — при некоректних	Passed	Passed
2	високий	Реєстрація нового користувача	Перехід на сторінку реєстрації, перевірка введених даних, створення акаунта	Passed	Passed
3	високий	Валідація полів при вході та реєстрації	Виведення повідомлень про помилки при порожніх або некоректних даних	Passed	Passed
4	високий	Генерація унікального бонусного коду	Код бонусної картки генерується та перевіряється на унікальність	Passed	Passed
5	високий	Завантаження головної сторінки	Відображення кнопки «Всі товари», списку категорій та блоку «Популярні товари»	Passed	Passed
6	високий	Перехід до списку всіх товарів	Відкривається сторінка з товарами, пошуком, кнопкою фільтра та списком усіх позицій	Passed	Passed
7	середній	Фільтрація товарів	Товари відфільтровуються за категоріями, ціною та гостротою	Passed	Passed

№	Пріоритет	Перевірка	Очікуваний результат	Windows 10	Windows 8
8	середній	Відкриття картки товару	Відображається повна інформація про товар, кнопки +/- та «Додати в кошик»	Passed	Passed
9	високий	Додавання товару в кошик	Товар додається до кошика з перевіркою кількості на складі	Passed	Passed
10	високий	Керування бонусами при замовленні	Валідація бонусів, списання або нарахування на картку клієнта	Passed	Passed
11	середній	Відображення історії замовлень у профілі	Список оформлених замовлень, відсортований за датою	Passed	Passed
12	низький	Перегляд деталей замовлення	Відображення списку товарів у вибраному замовленні	Passed	Passed
13	середній	Завершення сесії користувача	Кнопка «Вийти» очищує дані авторизації та повертає на екран входу	Passed	Passed

Тестування мобільного додатку KoreaShop на пристроях з Android 13 та Android 10 підтвердило стабільну та коректну роботу основного функціоналу. Додаток успішно виконував обробку даних, коректно реагував на введення користувачем і виявив повну сумісність з обома версіями операційної системи без збоїв чи критичних помилок.

ВИСНОВОК

У процесі розробки мобільного клієнтського додатку для мережі магазинів «Смак Кореї» було підтверджено актуальність потреби в автоматизованій, зручній та доступній системі для взаємодії клієнтів із торговою мережею. Аналіз сучасних рішень та конкурентного середовища показав, що мобільні додатки з гнучким функціоналом, підтримкою бонусних програм і можливістю швидкого оформлення замовлень стали стандартом у роздрібній торгівлі. Відсутність такого інструменту в компанії створювала бар'єри для зростання клієнтської лояльності та знижувала конкурентоздатність.

Розроблений додаток забезпечує користувачеві повноцінний набір функцій: зручну авторизацію, реєстрацію з перевіркою введених даних, фільтрацію товарів, перегляд детальної інформації про продукти, додавання до кошика, списання або накопичення бонусів, оформлення замовлення та перегляд історії покупок. Інтерфейс адаптований до мобільних пристроїв, логічно структурований і легко зрозумілий навіть для нових користувачів.

Рішення реалізовано з використанням сучасного технологічного стеку: .NET MAUI для кросплатформенної розробки мобільного застосунку, ASP.NET Core Web API для обробки запитів та Microsoft SQL Server для зберігання даних. Розробка здійснювалась у Visual Studio, що забезпечило швидкість, стабільність і масштабованість системи. Проведене тестування показало стабільну роботу додатку на пристроях із різними версіями Android (10 та 13), повну функціональну відповідність і відсутність критичних помилок. Усі ключові сценарії — від реєстрації до успішного оформлення замовлення — пройшли перевірку з позитивним результатом.

Таким чином, розроблена система:

- задовольняє поточні потреби користувачів у зручному доступі до товарів та персональних даних
- автоматизує ключові дії клієнта, скорочуючи час на оформлення замовлення;
- стимулює повторні покупки завдяки інтегрованій бонусній програмі;

- підвищує лояльність до бренду та створює цифрову основу для майбутнього розвитку.

Впровадження мобільного додатку є стратегічно доцільним рішенням, що дозволяє «Смаку Кореї» не лише відповідати очікуванням сучасного покупця, але й посилювати свої позиції на ринку через інноваційний підхід до обслуговування клієнтів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Методичні рекомендації до виконання кваліфікаційної роботи на здобуття освітнього ступеня «бакалавр» спеціальності 122 «Комп'ютерні науки» освітньо-професійної програми «Інформаційні системи та штучний інтелект» денної форми здобуття освіти [Електрон. ресурс] / уклад. С. В. Грибков, Н. В. Лімaнська, М. П. Костіков. – К.: НУХТ, 2025. – 43 с.
2. Смак Кореї – Taste of Korea – Корейські продукти в Україні [Електронний ресурс]. – Режим доступу: https://www.smak-korea.com.ua/?srsltid=AfmBOoqIR44-5I-mV9hrD6yIOAy7Wt21b-0foq8FmWbILQEqdwa3tKTV (дата звернення: 30.04.2025). – Назва з екрана.
3. ASIA FOODS: Магазин продуктів з Азії [Електронний ресурс]. – Режим доступу: https://asiafoods.com.ua/?srsltid=AfmBOorsb13XPnbWgprfs_mZ5OHX5ObjVOodj1XykFkG7JC9W3_Fapfg (дата звернення: 30.04.2025). – Назва з екрана.
4. SAMGUK – Korean Products [Електронний ресурс]. – Режим доступу: https://samguk.com.ua/products/sale_nabori/ (дата звернення: 30.04.2025). – Назва з екрана.
5. CA ERWin Data Modeler (ERWin) [Електронний ресурс]. – Режим доступу: https://stud.com.ua/77235/informatika/erwin_data_modeler_erwin (дата звернення: 30.04.2023). – Назва з екрана.
6. .NET Multi-platform App UI documentation [Електронний ресурс]. – Режим доступу: https://learn.microsoft.com/uk-ua/dotnet/maui/?view=net-maui-9.0 (дата звернення: 05.05.2025). – Назва з екрана.
7. Microsoft SQL documentation [Електронний ресурс]. – Режим доступу: https://learn.microsoft.com/uk-ua/sql/?view=sql-server-ver16 (дата звернення: 05.05.2025). – Назва з екрана.
8. Create a controller-based web API with ASP.NET Core [Електронний ресурс]. – Режим доступу: https://learn.microsoft.com/uk-ua/aspnet/core/tutorials/first-web-api?view=aspnetcore-9.0&tabs=visual-studio (дата звернення: 05.05.2025). – Назва з екрана.

9. Visual Studio IDE documentation [Електронний ресурс]. – Режим доступу: https://learn.microsoft.com/uk-ua/visualstudio/ide/?view=vs-2022 (дата звернення: 05.05.2025). – Назва з екрана.
10. C# language documentation [Електронний ресурс]. – Режим доступу: https://learn.microsoft.com/uk-ua/dotnet/csharp/ (дата звернення: 05.05.2025). – Назва з екрана.
11. XAML tools documentation [Електронний ресурс]. – Режим доступу: https://learn.microsoft.com/uk-ua/visualstudio/xaml-tools/?view=vs-2022 (дата звернення: 05.05.2025). – Назва з екрана.
12. Android OS Documentation [Електронний ресурс]. – Режим доступу: https://source.android.com/docs (дата звернення: 05.05.2025). – Назва з екрана.
13. Бази даних в комп'ютерних мережах [Електронний ресурс]. – Режим доступу: https://stud.com.ua/62416/menedzhment/bazi_danih_kompyuternih_merezhah (дата звернення: 05.05.2025). – Назва з екрана.
14. Swagger Documentation | Swagger Docs [Електронний ресурс]. – Режим доступу: https://swagger.io/docs/ (дата звернення: 05.05.2025). – Назва з екрана.
15. Комплексний інструмент автоматизації розробки API Swagger [Електронний ресурс]. – Режим доступу: https://freehost.com.ua/ukr/faq/articles/swagger-scho-tse-take-ta-jak-z-nim-pratsjuvati/ (дата звернення: 05.05.2025). – Назва з екрана.
16. Проектування графічного призначеного для користувача інтерфейсу [Електронний ресурс]. – Режим доступу: https://stud.com.ua/174174/informatika/proektuvannya_grafichnogo_priznachenogo_koristuvacha_interfeysu (дата звернення: 05.05.2025). – Назва з екрана.
17. МОБІЛЬНІ ДОДАТКИ ДЛЯ ТОРГІВЛІ [Електронний ресурс]. – Режим доступу: https://stud.com.ua/84307/ekonomika/mobilni_dodatki_torgivli (дата звернення: 05.05.2025). – Назва з екрана.
18. Інформаційні системи мобільної торгівлі [Електронний ресурс]. – Режим доступу: https://stud.com.ua/20602/informatika/informatsiyni_sistemi_mobilnoyi_torgivli (дата звернення: 05.05.2025). – Назва з екрана.

19. ДСТУ ISO/IEC/IEEE 29119-1:2017. Інженерія систем і програмних засобів тестування програмних засобів. На заміну НА ЗАМІНУ ДСТУ ISO/IEC/IEEE 29119-1:2015 ; чинний від 19.12.2017. Київ : Не є офіційним виданням., 2018. 20 с. (дата звернення: 05.05.2025).
20. ДСТУ 3008:2015. ЗВІТИ У СФЕРІ НАУКИ І ТЕХНІКИ. На заміну ДСТУ 3008-95 ; чинний від 22.06.2015. Київ : ДП «УкрНДНЦ», 2016. 26 с. URL: https://science.kname.edu.ua/images/dok/derzhstandart_3008_2015.pdf (дата звернення: 05.05.2025).
21. Design for Android [Електронний ресурс]. – Режим доступу: https://developer.android.com/design/ui (дата звернення: 05.05.2025). – Назва з екрана.
22. Documentation - erwin Data Modeler [Електронний ресурс]. – Режим доступу: https://bookshelf.erwin.com/bookshelf/public_html/2020R1/Content/Release%20Notes/Data%20Modeler%20Release%20Notes/6940.html (дата звернення: 05.05.2025). – Назва з екрана.
23. HTTP documentation [Електронний ресурс]. – Режим доступу: https://devdocs.io/http/ (дата звернення: 05.05.2025). – Назва з екрана.
24. Принципи створення адаптивного дизайну для інтернет-магазинів [Електронний ресурс]. – Режим доступу: https://web.dev/responsive-web-design-basics/ (дата звернення: 05.05.2025). – Назва з екрана.
25. A Complete Guide to Mobile App Development [Електронний ресурс]. – Режим доступу: https://buildfire.com/understanding-mobile-app-development-lifecycle/ (дата звернення: 05.05.2025). – Назва з екрана.
26. Asp.Net Core 9 (.NET 9) | True Ultimate Guide [Електронний ресурс]. – Режим доступу: https://www.udemy.com/course/asp-net-core-true-ultimate-guide-real-project/?srsltid=AfmBOopm7kU44tXcmo35P1J9muVUxIEl0umSl5BXcR74t3Si0NyG1A5w (дата звернення: 05.05.2025). – Назва з екрана.
27. MAUI Development Guide | Documentation Center [Електронний ресурс]. – Режим доступу: https://docs.aspnetzero.com/aspnet-core-

mvc/latest/Development-Guide-MAUI (дата звернення: 05.05.2025). – Назва з екрана.

28. NET MAUI - Android Project [Електронний ресурс]. – Режим доступу: https://documentation.anyline.com/dotnet-sdk-component/latest/maui/maui_implementation_android.html (дата звернення: 05.05.2025). – Назва з екрана.
29. The Ultimate Guide to An Effective UI Design [Електронний ресурс]. – Режим доступу: https://www.uxpin.com/studio/blog/ui-design-guide/ (дата звернення: 05.05.2025). – Назва з екрана.

ДОДАТКИ

Додаток А. Схема організаційної структури мережі магазинів «Смак Кореї»

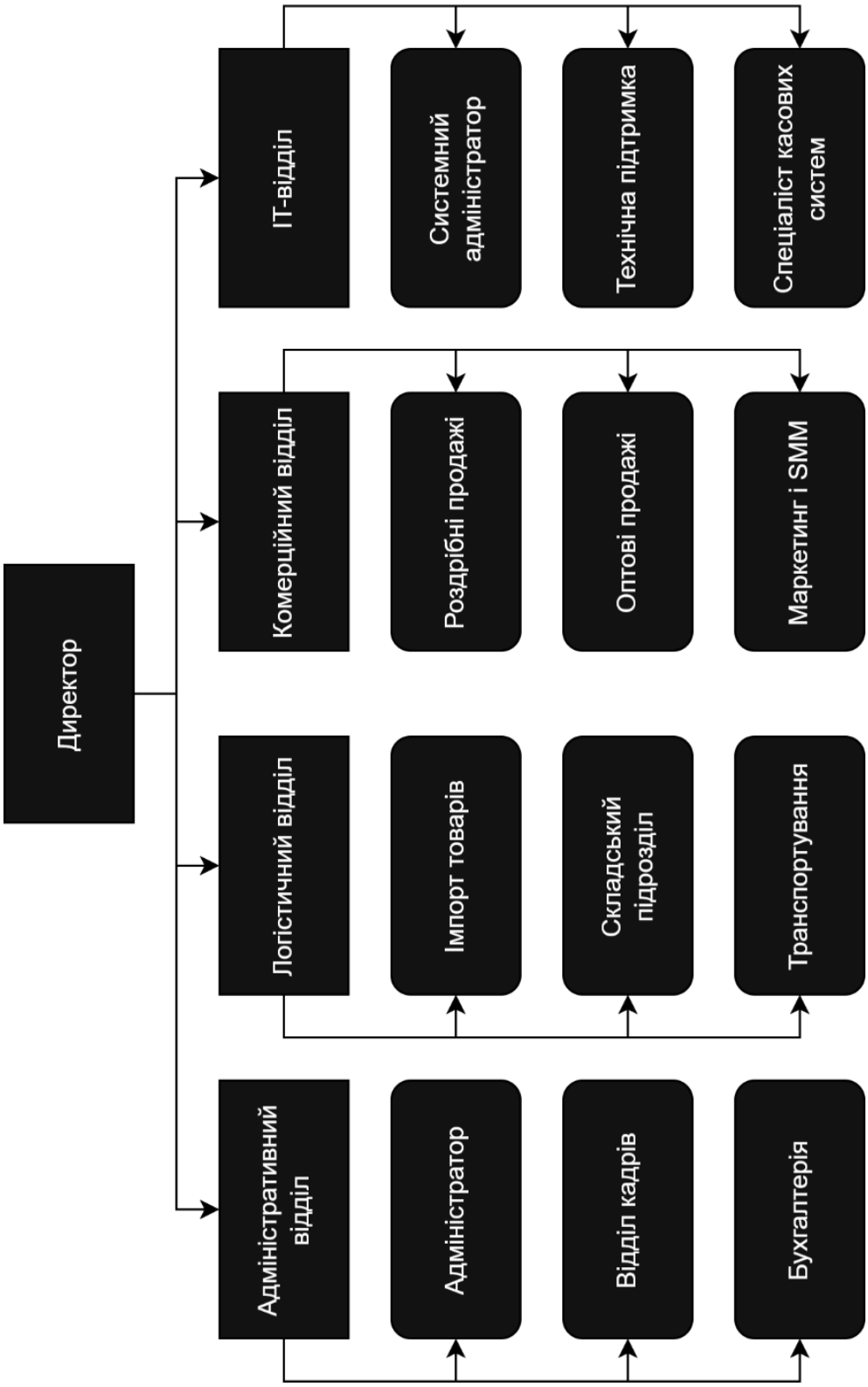

Рисунок А.1 – Структура підприємства

Додаток Б. Модель бізнес-процесів у нотації BPMN

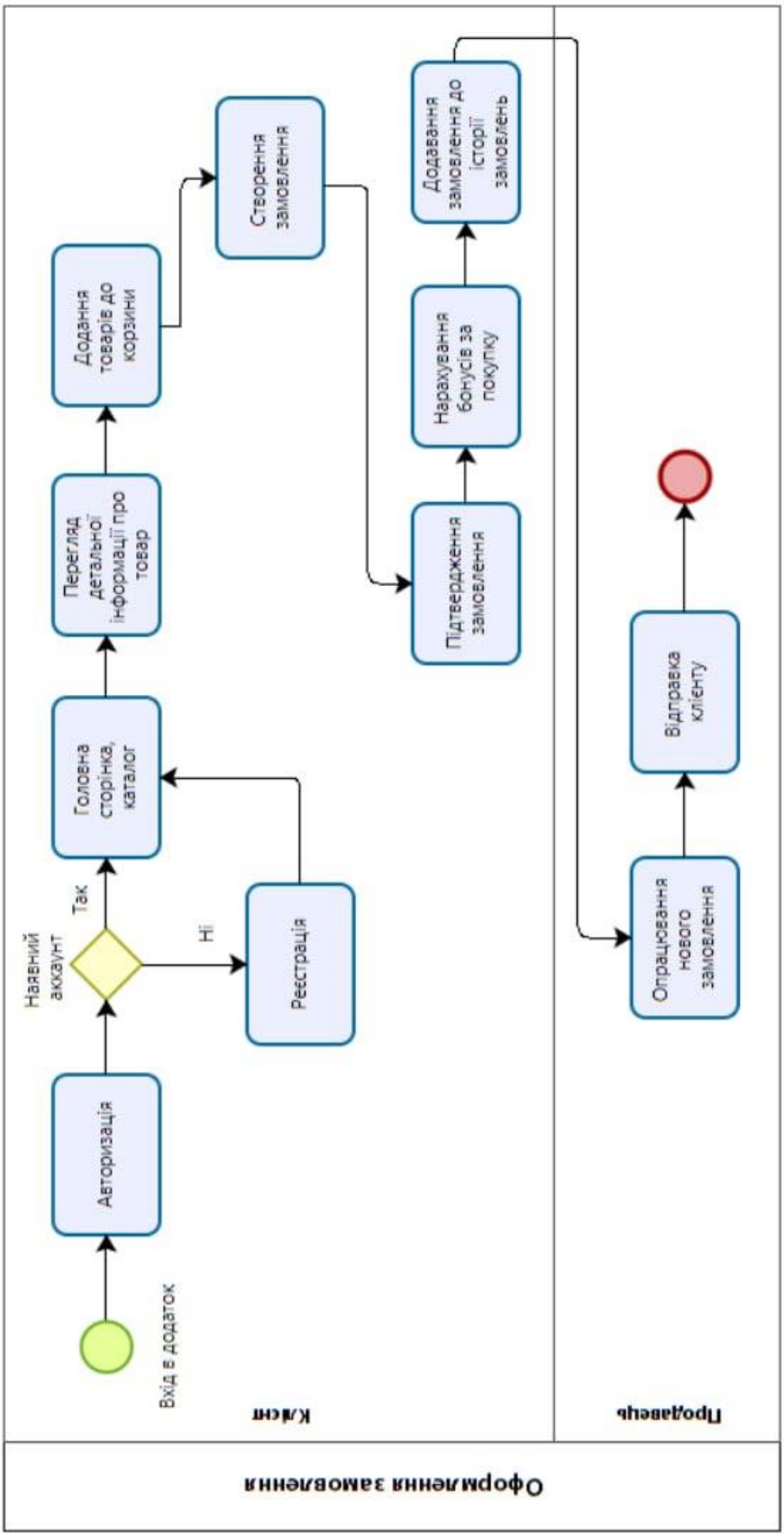

Рисунок Б.1 – Модель бізнес-процесів у нотації BPMN рівня TO-BE

Додаток В. Структура бази даних

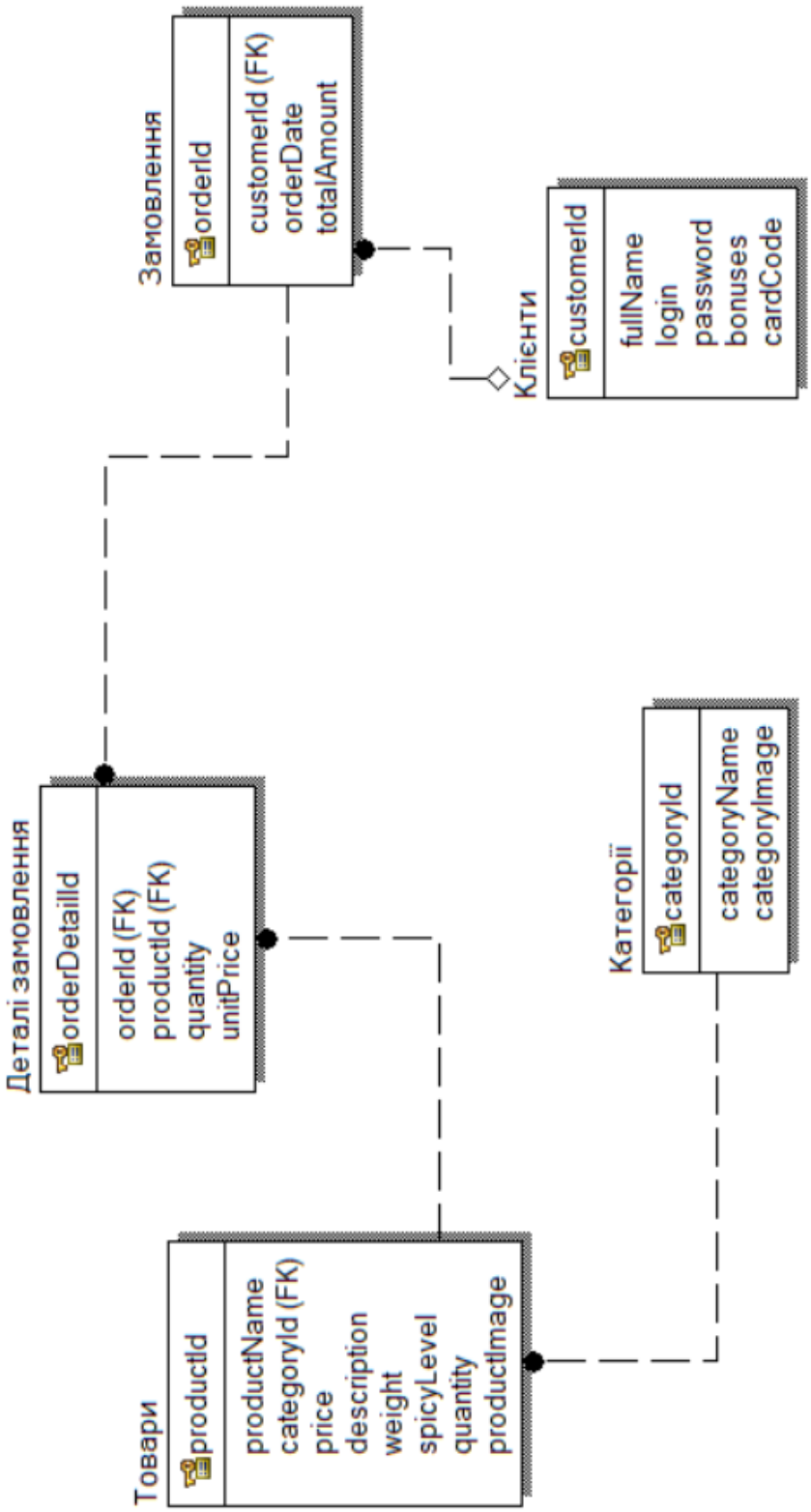

Рисунок В.1 – Логічна модель бази даних

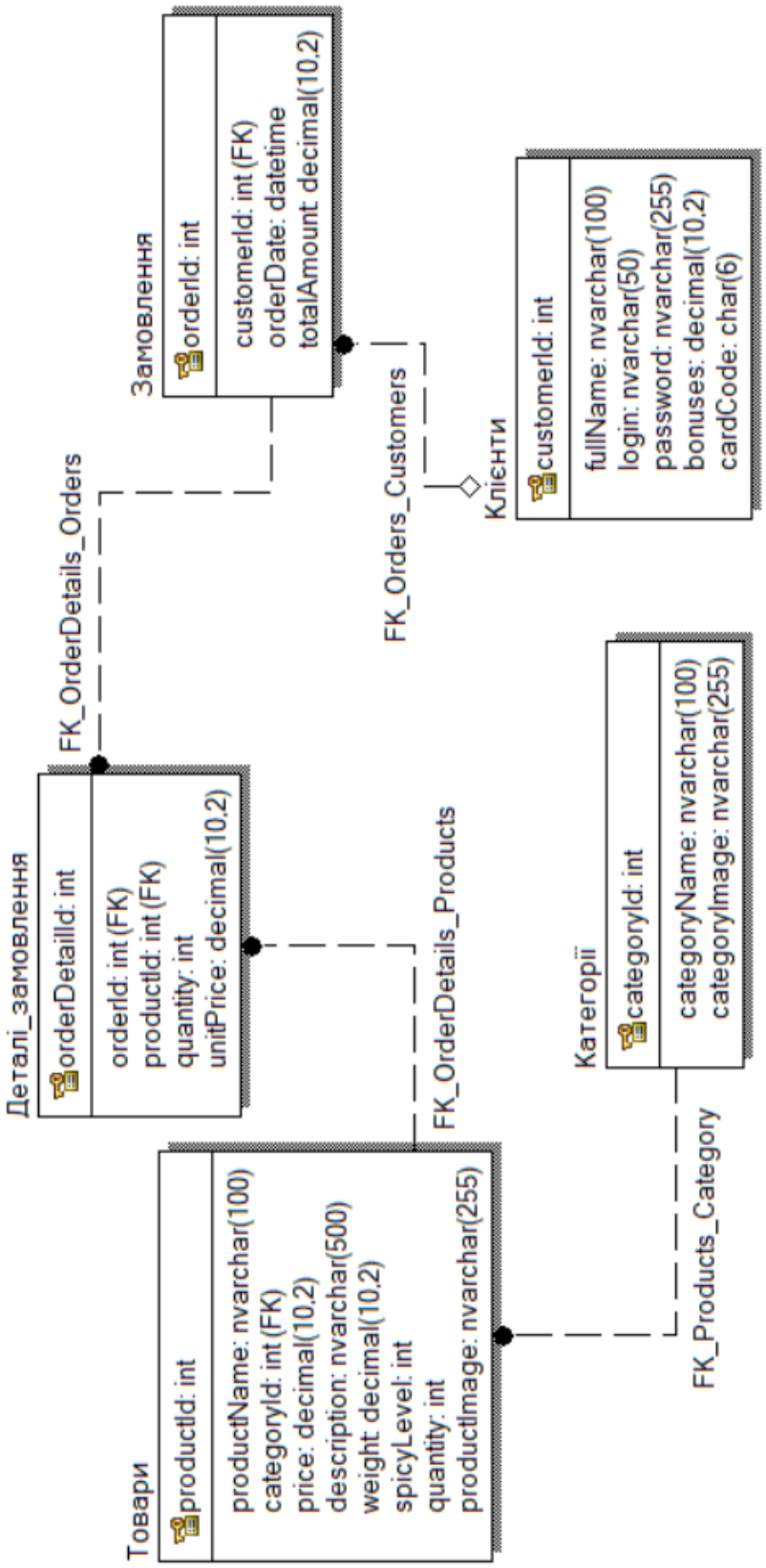

Рисунок В.2 – Фізична модель бази даних

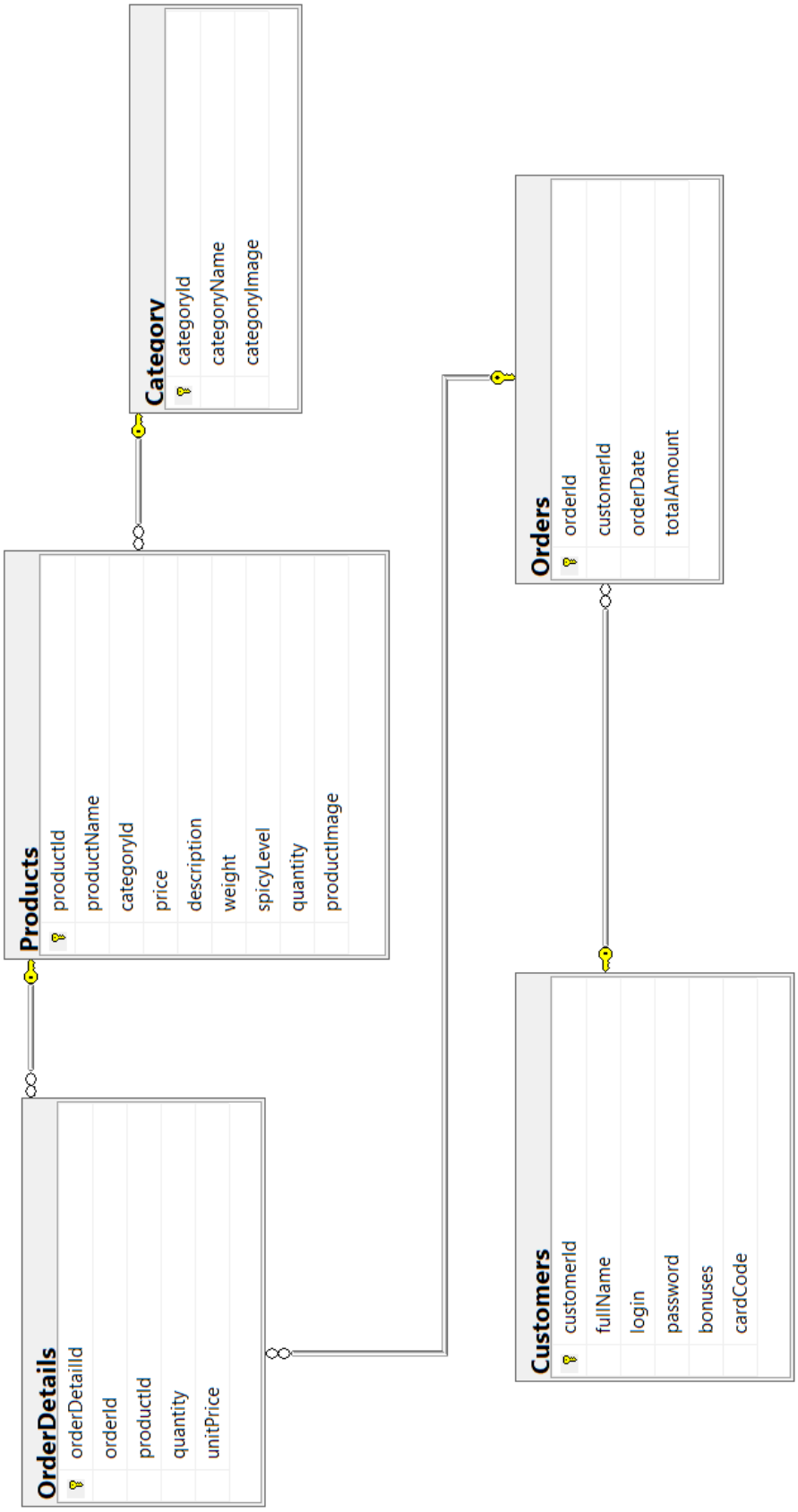

Рисунок В.3 – Модель бази даних в MS SQL Server

Додаток Г. Інтерфейс користувача

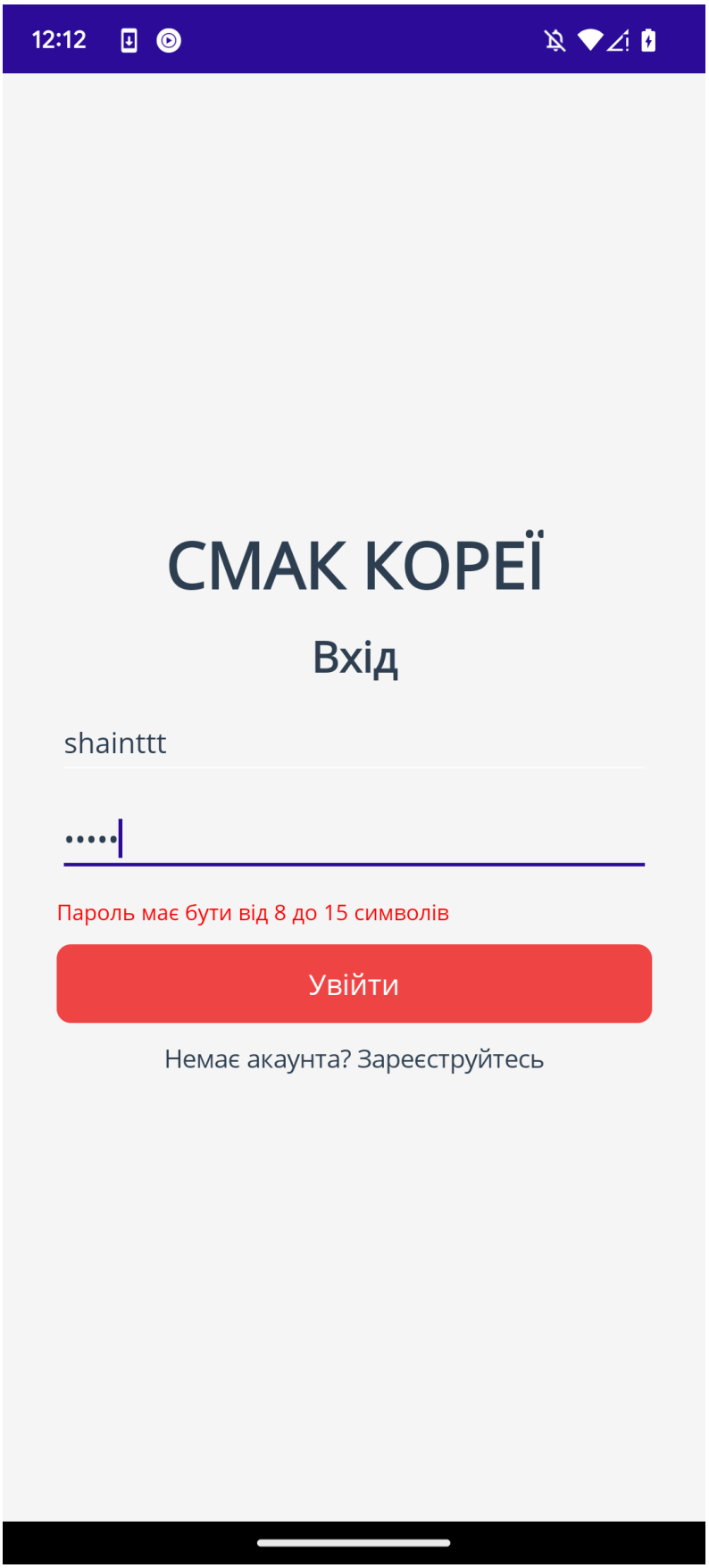

Рисунок Г.1 – Сторінка авторизації з перевіркою даних

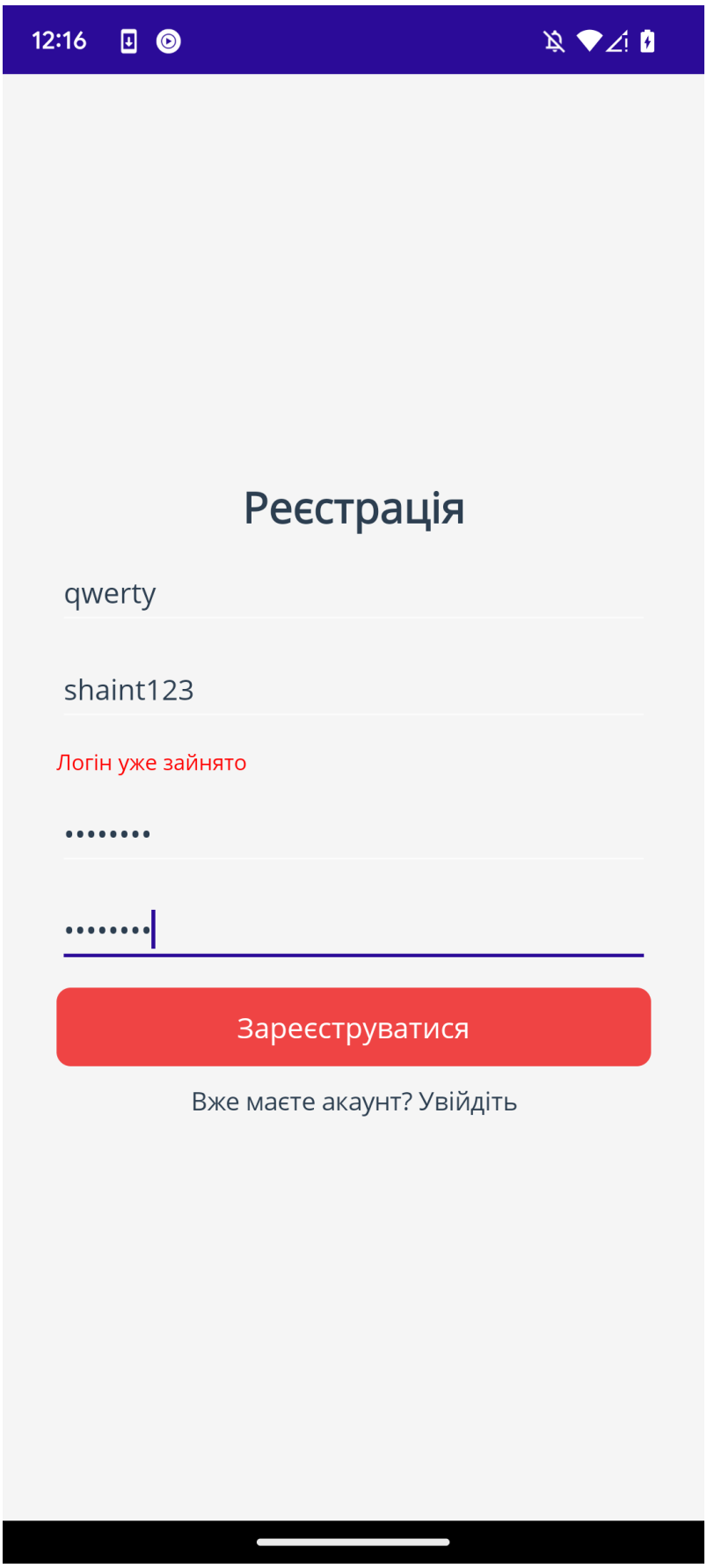

Рисунок Г.2 – Сторінка реєстрації з перевіркою даних

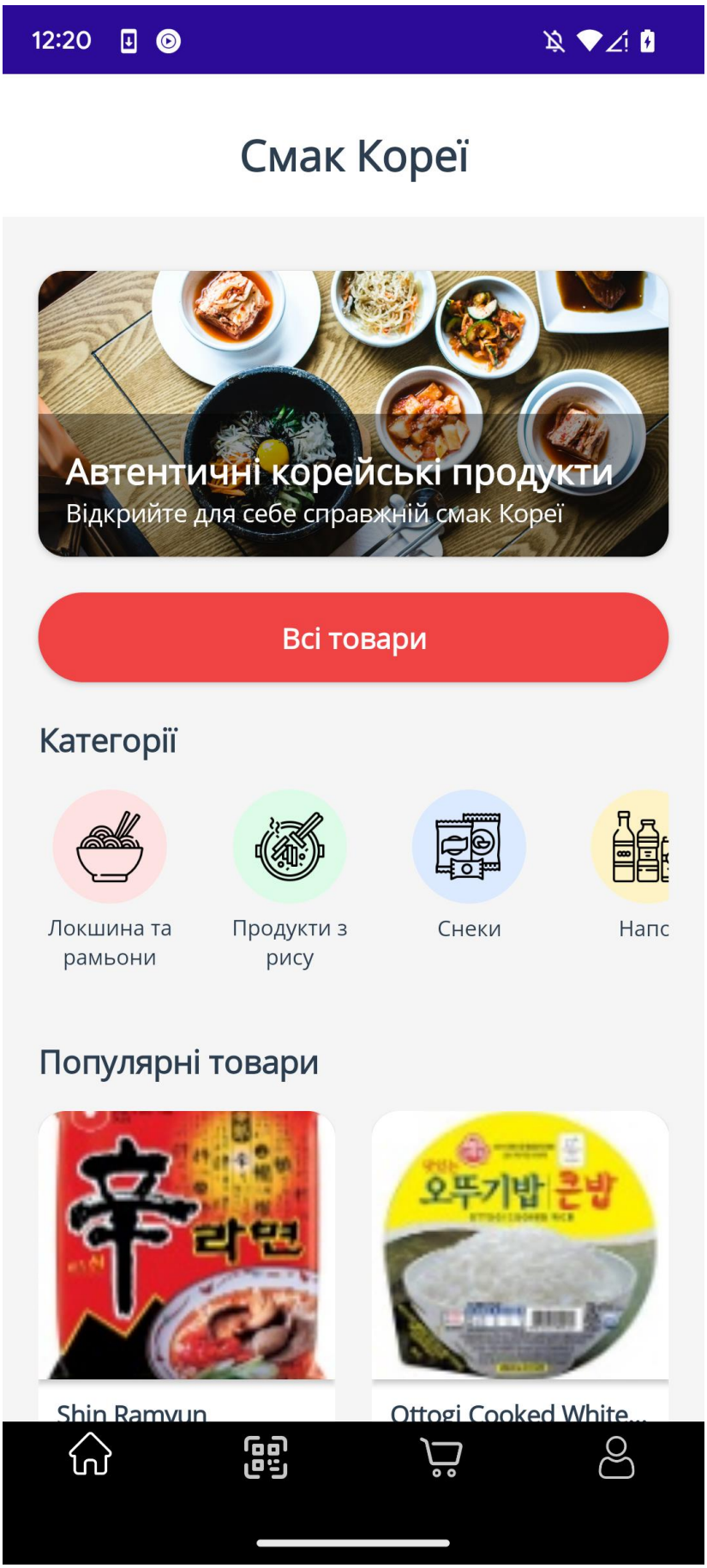

Рисунок Г.3 – Головна сторінка додатку

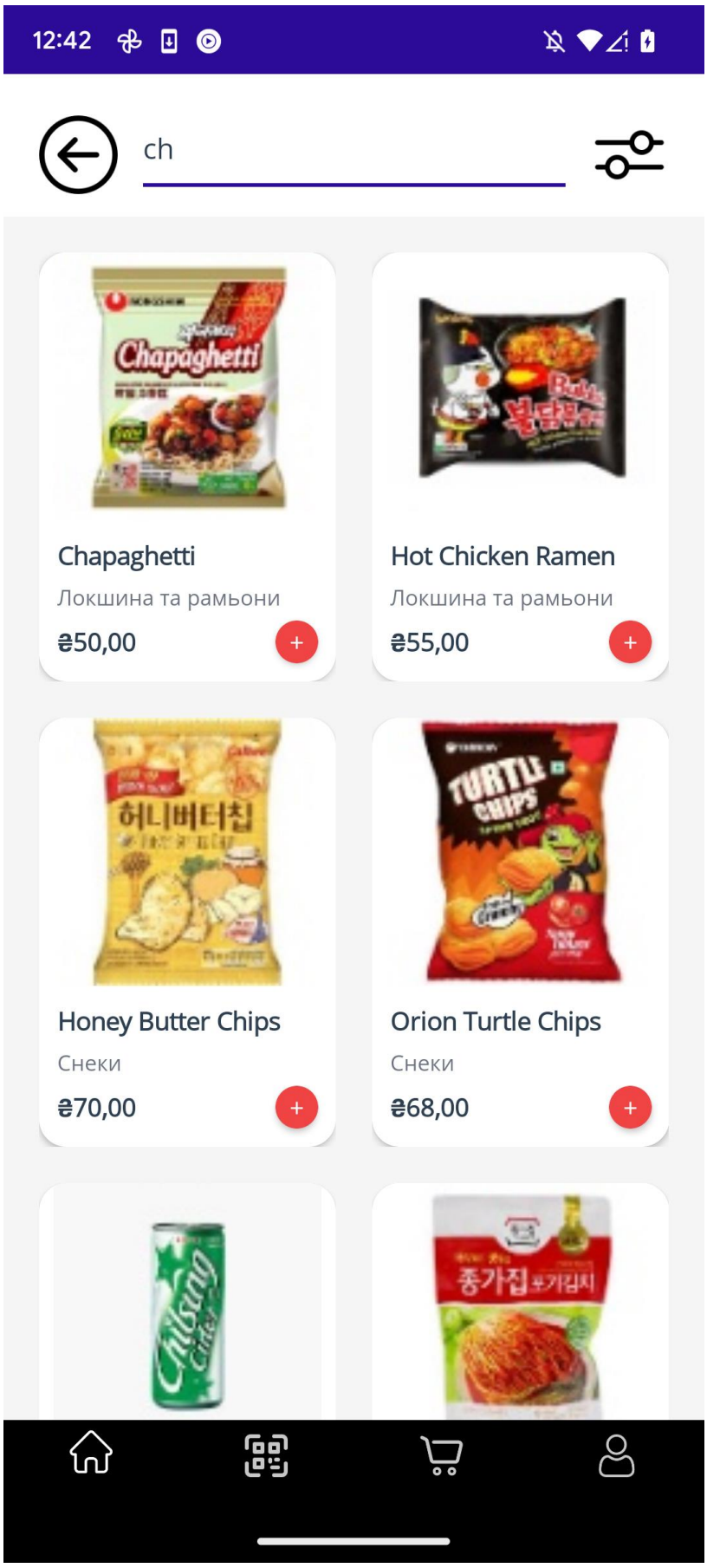

Рисунок Г.4 – Сторінка всі товару з фільтрацією по назві

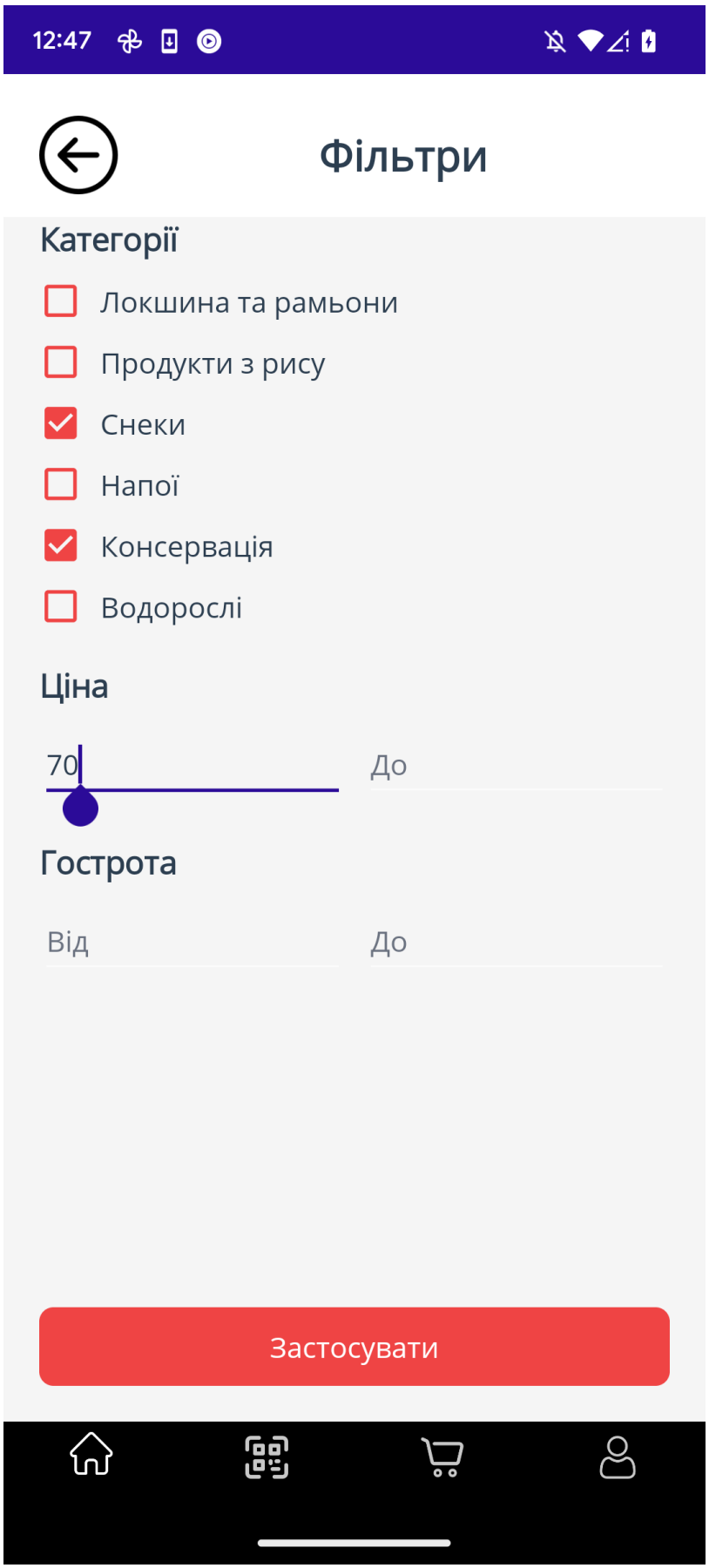

Рисунок Г.5 – Сторінка фільтрації

Рисунок Г.6 – Картка товару

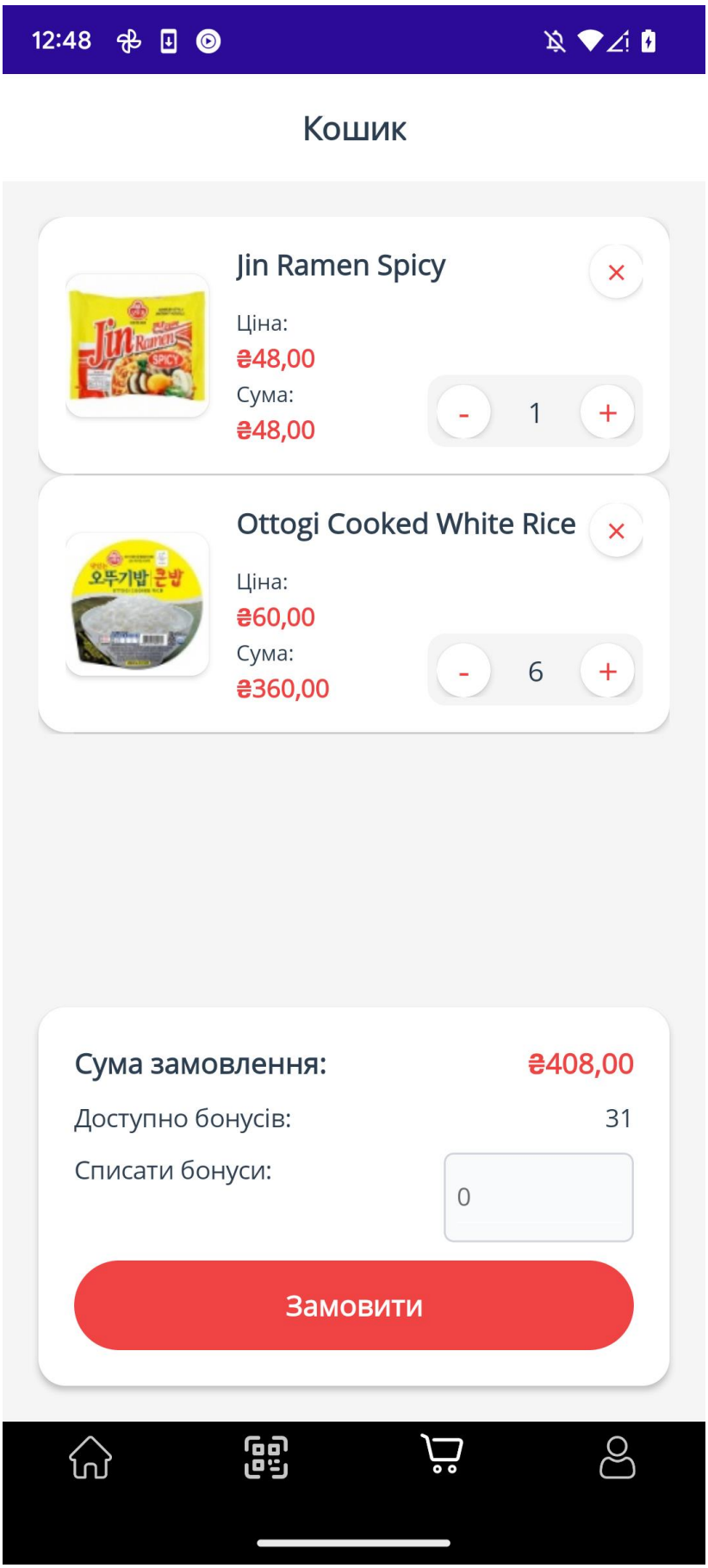

Рисунок Г.7 – Кошик з доданим товаром

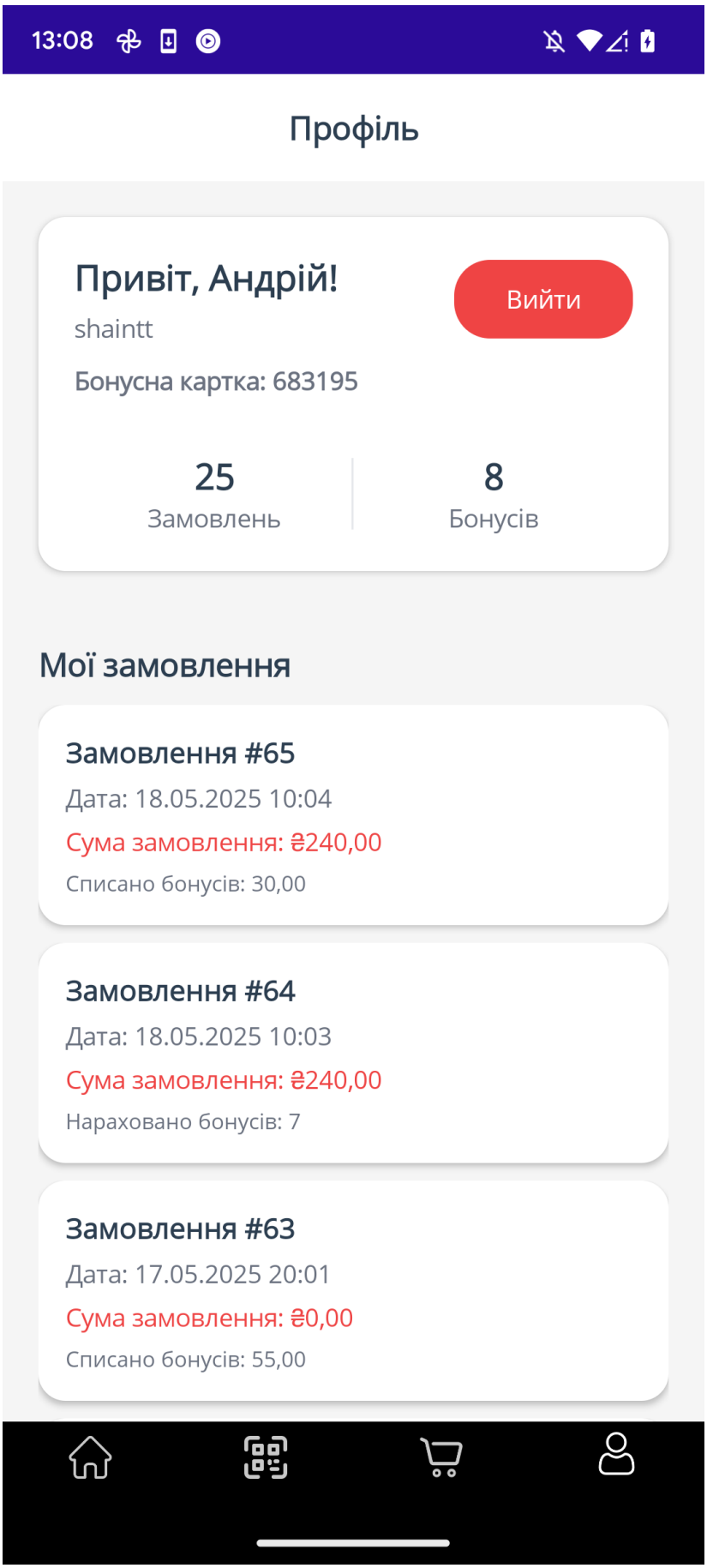

Рисунок Г.8 – Сторінка профіль користувача

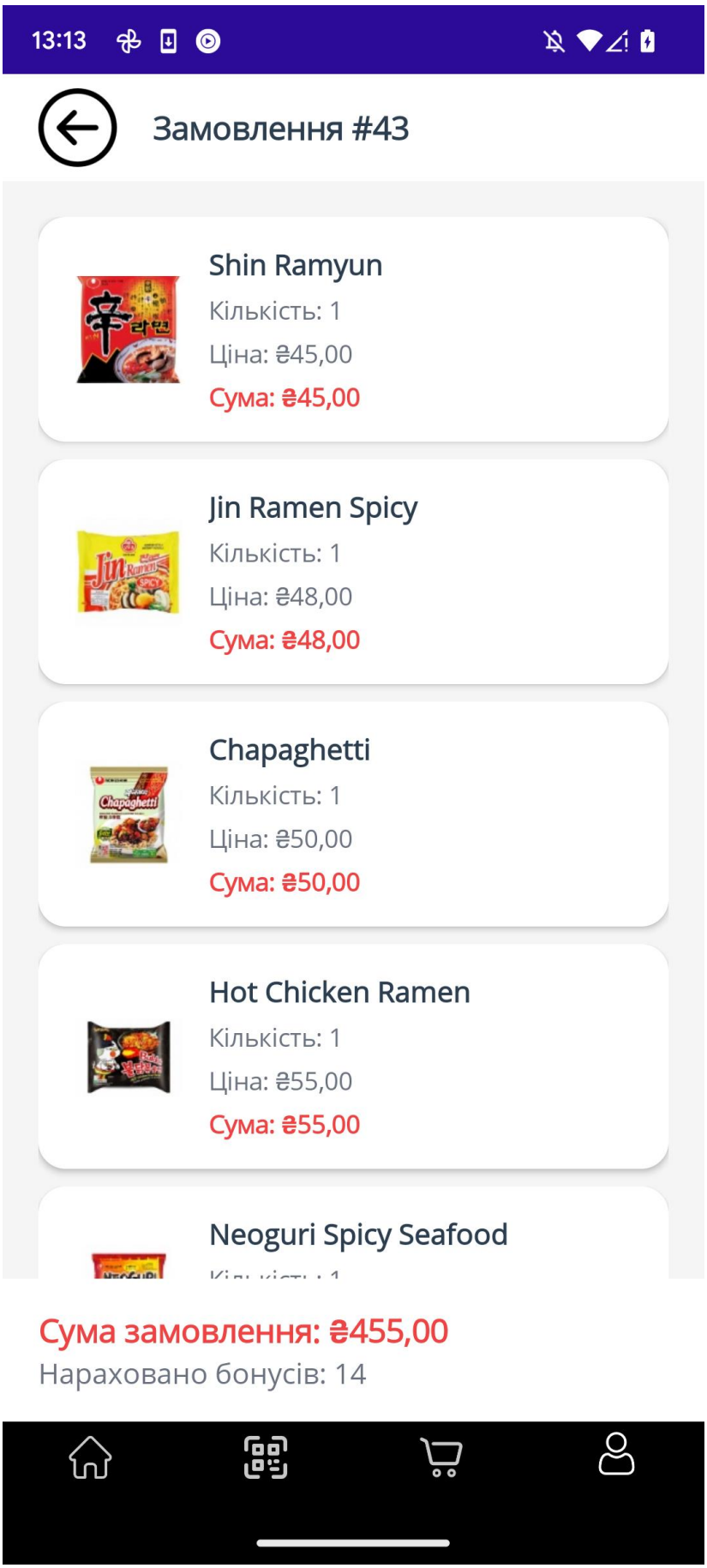

Рисунок Г.9 – Сторінка перегляду оформленого замовлення

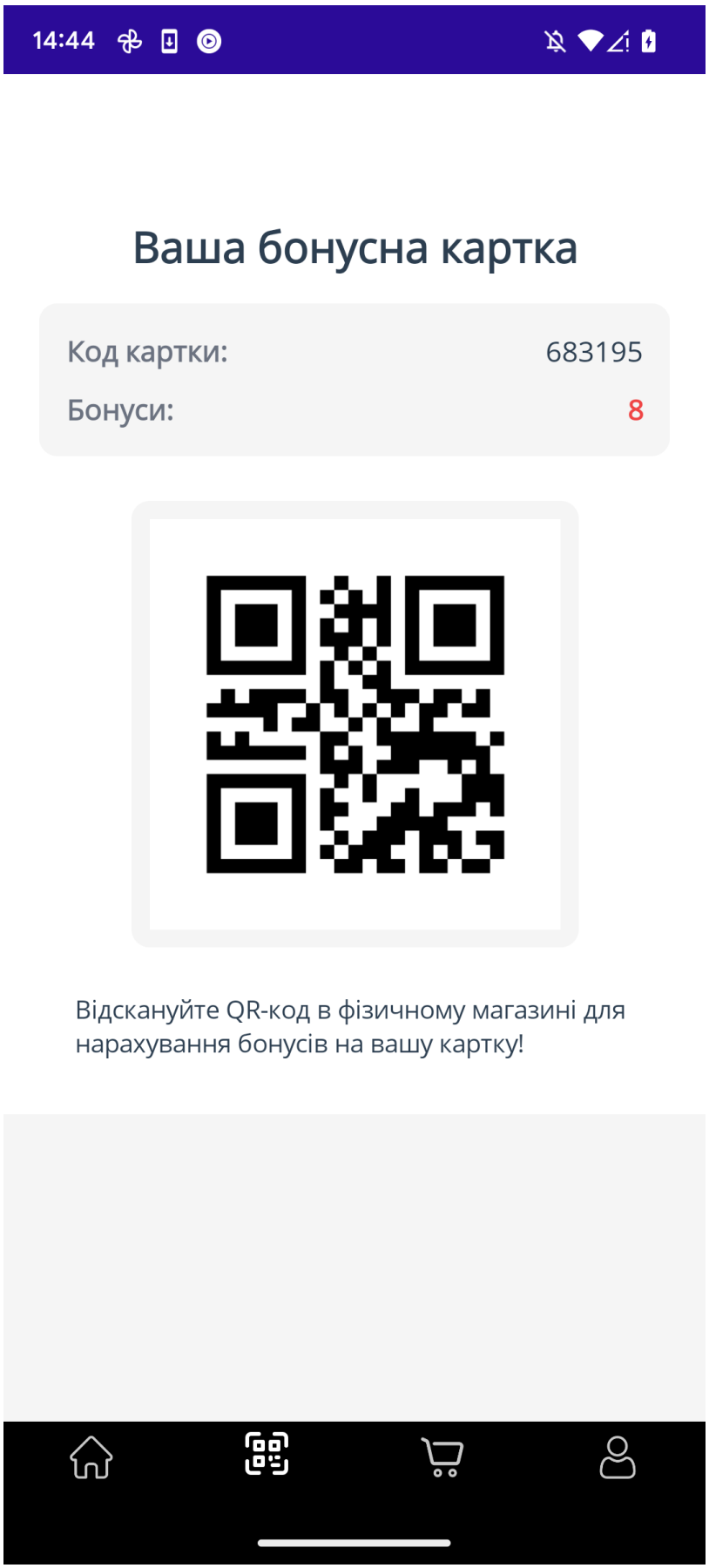

Рисунок Г.10 – Сторінка бонусної картки зі згенерованим QR-кодом

Додаток Д. Код програми

Д.1 – Сторінка авторизації

```
using KoreaShop.Services;
using Microsoft.Extensions.Logging.Abstractions;
using System;
using KoreaShop.Models;

namespace KoreaShop.Pages;

public partial class LoginPage : ContentPage
{
    private readonly ApiService _apiService;

    public LoginPage()
    {
        InitializeComponent();
        _apiService = new ApiService();
    }

    private async void LoginButton_Clicked(object sender, EventArgs e)
    {
        LoginErrorLabel.IsVisible = false;
        PasswordErrorLabel.IsVisible = false;
        InvalidPasswordLabel.IsVisible = false;

        var login = LoginEntry.Text?.Trim();
        var password = PasswordEntry.Text?.Trim();

        if (string.IsNullOrEmpty(login))
        {
            LoginErrorLabel.Text = "Введіть логін";
            LoginErrorLabel.IsVisible = true;
            return;
        }
        if (string.IsNullOrEmpty(password))
        {
            PasswordErrorLabel.Text = "Введіть пароль";
            PasswordErrorLabel.IsVisible = true;
            return;
        }

        if (login.Length < 5 || login.Length > 15)
        {
```

```
            LoginErrorLabel.Text = "Логін має бути від 5 до 15 символів";
            LoginErrorLabel.IsVisible = true;
            return;
        }
        if (password.Length < 8 || password.Length > 15)
        {
            PasswordErrorLabel.Text = "Пароль має бути від 8 до 15 символів";
            PasswordErrorLabel.IsVisible = true;
            return;
        }

        try
        {
            var isAuthenticated = await _apiService.AuthenticateAsync(login, password);
            if (isAuthenticated)
            {
                var customer = await _apiService.GetCustomerByLoginAsync(login);
                if (customer != null)
                {

                    Preferences.Set("CustomerId", customer.CustomerId);
                    Preferences.Set("FullName", customer.FullName);
                    Preferences.Set("Login", customer.Login);
                    Preferences.Set("Bonuses", customer.Bonuses.ToString());
                    Preferences.Set("CardCode", customer.CardCode);


                    SessionManager.CustomerId = customer.CustomerId;
                    SessionManager.Login = customer.Login;
                    SessionManager.Bonuses = customer.Bonuses;

                    Console.WriteLine($"SessionManager initialized:
CustomerId={SessionManager.CustomerId}, Login={SessionManager.Login},
Bonuses={SessionManager.Bonuses}");
                }
                else
                {
                    throw new Exception("Не вдалося отримати дані користувача");
                }
                Application.Current.MainPage = new AppShell();
            }
            else
            {
                InvalidPasswordLabel.IsVisible = true;
            }
```

```
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Login error: {ex.Message}");
            InvalidPasswordLabel.Text = "Невірний пароль або помилка сервера";
            InvalidPasswordLabel.IsVisible = true;
        }
    }

    private async void OnRegisterTapped(object sender, EventArgs e)
    {
        await Navigation.PushAsync(new RegisterPage());
    }

}
```

Д.2 – Сторінка реєстрації

```
namespace KoreaShop.Pages;
using KoreaShop.Services;

public partial class RegisterPage : ContentPage
{
    private readonly ApiService _apiService;

    public RegisterPage()
    {
        InitializeComponent();
        _apiService = new ApiService();
    }

    private async void RegisterButton_Clicked(object sender, EventArgs e)
    {
        FullNameErrorLabel.IsVisible = false;
        LoginErrorLabel.IsVisible = false;
        PasswordErrorLabel.IsVisible = false;
        ConfirmPasswordErrorLabel.IsVisible = false;

        var fullName = FullNameEntry.Text?.Trim();
        var login = LoginEntry.Text?.Trim();
        var password = PasswordEntry.Text?.Trim();
        var confirmPassword = ConfirmPasswordEntry.Text?.Trim();

        if (string.IsNullOrEmpty(fullName))
        {
            FullNameErrorLabel.Text = "Введіть ім'я";
```

```
    FullNameErrorLabel.IsVisible = true;
    return;
}
if (string.IsNullOrEmpty(login))
{
    LoginErrorLabel.Text = "Введіть логін";
    LoginErrorLabel.IsVisible = true;
    return;
}
if (string.IsNullOrEmpty(password))
{
    PasswordErrorLabel.Text = "Введіть пароль";
    PasswordErrorLabel.IsVisible = true;
    return;
}
if (string.IsNullOrEmpty(confirmPassword))
{
    ConfirmPasswordErrorLabel.Text = "Повторіть пароль";
    ConfirmPasswordErrorLabel.IsVisible = true;
    return;
}

if (fullName.Length < 2 || fullName.Length > 25)
{
    FullNameErrorLabel.Text = "Ім'я має бути від 2 до 25 символів";
    FullNameErrorLabel.IsVisible = true;
    return;
}
if (login.Length < 5 || login.Length > 15)
{
    LoginErrorLabel.Text = "Логін має бути від 5 до 15 символів";
    LoginErrorLabel.IsVisible = true;
    return;
}
if (password.Length < 8 || password.Length > 15)
{
    PasswordErrorLabel.Text = "Пароль має бути від 8 до 15 символів";
    PasswordErrorLabel.IsVisible = true;
    return;
}

if (password != confirmPassword)
{
    ConfirmPasswordErrorLabel.Text = "Паролі не збігаються";
    ConfirmPasswordErrorLabel.IsVisible = true;
```

```
            return;
        }

        try
        {
            var isLoginUnique = await _apiService.CheckLoginUniqueAsync(login);
            if (!isLoginUnique)
            {
                LoginErrorLabel.Text = "Логін уже зайнято";
                LoginErrorLabel.IsVisible = true;
                return;
            }

            string cardCode = null;
            int maxAttempts = 10;
            for (int i = 0; i < maxAttempts; i++)
            {
                cardCode = GenerateCardCode();
                var isCardCodeUnique = await
_apiService.CheckCardCodeUniqueAsync(cardCode);
                if (isCardCodeUnique)
                    break;
                cardCode = null;
            }

            if (cardCode == null)
            {
                LoginErrorLabel.Text = "Не вдалося згенерувати код бонусної карти";
                LoginErrorLabel.IsVisible = true;
                return;
            }

            var isRegistered = await _apiService.RegisterAsync(fullName, login,
password, cardCode);
            if (isRegistered)
            {
                await Navigation.PopAsync();
            }
            else
            {
                LoginErrorLabel.Text = "Не вдалося зареєструвати користувача";
                LoginErrorLabel.IsVisible = true;
            }
        }
        catch (Exception ex)
```

```
        {
            Console.WriteLine($"Registration error: {ex.Message}");
            LoginErrorLabel.Text = "Помилка реєстрації";
            LoginErrorLabel.IsVisible = true;
        }
    }

    private async void OnLoginTapped(object sender, EventArgs e)
    {
        await Navigation.PopAsync();
    }

    private string GenerateCardCode()
    {
        var random = new Random();
        return random.Next(100000, 999999).ToString();
    }
}
```

Д.3 – Головна сторінка

```
using KoreaShop.Services;
using KoreaShop.Models;
using System.Collections.ObjectModel;
using System.Linq;
using System.Windows.Input;

namespace KoreaShop.Pages;

public partial class HomePage : ContentPage
{
    private readonly ApiService _apiService;
    public ObservableCollection<CategoryViewModel> Categories { get; set; }
    public ObservableCollection<ProductViewModel> PopularProducts { get; set; }

    public HomePage()
    {
        InitializeComponent();
        _apiService = new ApiService();

        Categories = new ObservableCollection<CategoryViewModel>();
        PopularProducts = new ObservableCollection<ProductViewModel>();

        BindingContext = this;
        LoadData();
    }
```

```
    private async void LoadData()
    {
        try
        {
            var categories = await _apiService.GetCategoriesAsync();
            Categories.Clear();
            foreach (var category in categories)
            {
                Categories.Add(new CategoryViewModel
                {
                    CategoryId = category.CategoryId,
                    Name = category.CategoryName,
                    Icon = category.CategoryImage ??
"https://example.com/default_icon.png",
                    FrameColor = GetCategoryFrameColor(category.CategoryName)
                });
            }

            var products = await _apiService.GetProductsAsync();
            PopularProducts.Clear();

            var popular = new List<ProductViewModel>();
            var categoryIds = categories.Select(c => c.CategoryId).ToList();

            foreach (var categoryId in categoryIds)
            {
                var product = products.FirstOrDefault(p => p.CategoryId == categoryId);
                if (product != null)
                {
                    popular.Add(new ProductViewModel
                    {
                        ProductId = product.ProductId,
                        Name = product.ProductName,
                        Description = product.Description,
                        Price = product.Price,
                        ImageUrl = product.ProductImage,
                        Quantity = product.Quantity,
                        CategoryId = product.CategoryId,
                        SpicyLevel = product.SpicyLevel,
                        Weight = product.Weight
                    });
                }
            }
```

```
        if (popular.Count < 4)
        {
            var remainingProducts = products
                .Where(p => !popular.Any(pp => pp.ProductId == p.ProductId))
                .Take(4 - popular.Count);
            foreach (var product in remainingProducts)
            {
                popular.Add(new ProductViewModel
                {
                    ProductId = product.ProductId,
                    Name = product.ProductName,
                    Description = product.Description,
                    Price = product.Price,
                    ImageUrl = product.ProductImage,
                    Quantity = product.Quantity,
                    CategoryId = product.CategoryId,
                    SpicyLevel = product.SpicyLevel,
                    Weight = product.Weight
                });
            }
        }

        foreach (var product in popular)
        {
            PopularProducts.Add(product);
        }
    }
    catch
    {
        await DisplayAlert("Помилка", "Не вдалося завантажити дані.", "OK");
    }
}

private string GetCategoryFrameColor(string categoryName)
{
    return categoryName switch
    {
        "Локшина та рамьони" => "#FEE2E2",
        "Продукти з рису" => "#DCFCE7",
        "Снеки" => "#DBEAFE",
        "Напої" => "#FEF3C7",
        "Консервація" => "#F3E8FF",
        "Водорості" => "#E6F0FA",
        _ => "#FFFFFF"
    };
```

```
    }

    private async void OnAllProductsTapped(object sender, EventArgs e)
    {
        await Shell.Current.GoToAsync(nameof(AllProductsPage));
    }

    private async void OnCategoryTapped(object sender, TappedEventArgs e)
    {
        if (e.Parameter is int categoryId)
        {
            await
Shell.Current.GoToAsync($"{nameof(AllProductsPage)}?CategoryId={categoryId}"
);
        }
    }

    private async void OnProductTapped(object sender, TappedEventArgs e)
    {
        if (e.Parameter is int productId)
        {
            await
Shell.Current.GoToAsync($"{nameof(ProductDetailPage)}?ProductId={productId}")
;
        }
    }

    private async void OnAddToCartTapped(object sender, TappedEventArgs e)
    {
        if (sender is Frame frame && frame.GestureRecognizers.FirstOrDefault() is
TapGestureRecognizer recognizer)
        {
            var product = recognizer.CommandParameter as ProductViewModel;
            if (product != null)
            {
                var cartService = CartService.Instance;
                var cartItem = cartService.CartItems.FirstOrDefault(item =>
item.Product.ProductId == product.ProductId);

                if (cartItem != null)
                {
                    int newQuantity = cartItem.Quantity + 1;
                    if (newQuantity <= product.Quantity)
                    {
                        cartItem.Quantity = newQuantity;
```

```
                    await DisplayAlert("Кошик", $"Кількість '{product.Name}'
оновлено до {newQuantity} шт.", "OK");
                }
                else
                {
                    await DisplayAlert("Увага", $"На складі залишилось тільки
{product.Quantity} одиниць товару.", "OK");
                    return;
                }
            }
            else
            {
                if (1 <= product.Quantity)
                {
                    cartService.CartItems.Add(new CartItemViewModel
                    {
                        Product = product,
                        Quantity = 1
                    });
                    await DisplayAlert("Кошик", $"Товар '{product.Name}' додано до
кошика у кількості 1 шт.", "OK");
                }
                else
                {
                    await DisplayAlert("Увага", $"На складі залишилось тільки
{product.Quantity} одиниць товару.", "OK");
                    return;
                }
            }

            MessagingCenter.Send(this, "CartUpdated");
        }
    }
  }
}
```

Д.4 – Сторінка всі товари

```
using KoreaShop.Services;
using KoreaShop.Models;
using System.Collections.ObjectModel;
using System.ComponentModel;
using System.Windows.Input;
using System.Runtime.CompilerServices;

namespace KoreaShop.Pages;
```

```
[QueryProperty(nameof(CategoryId), "CategoryId")]
public partial class AllProductsPage : ContentPage, INotifyPropertyChanged
{
    private readonly ApiService _apiService;
    private readonly ObservableCollection<ProductViewModel> _allProducts;
    private bool _isLoading;
    private string _lastSearchText;
    private FilterOptions _currentFilters;
    private Task _debounceTask;
    private Dictionary<int, string> _categoryNames;

    private int _categoryId;

    public int CategoryId
    {
        get => _categoryId;
        set
        {
            _categoryId = value;
            OnPropertyChanged();
            ApplyCategoryFilter();
        }
    }

    public ObservableCollection<ProductViewModel> FilteredProducts { get; set; }

    public bool IsLoading
    {
        get => _isLoading;
        set
        {
            if (_isLoading != value)
            {
                _isLoading = value;
                OnPropertyChanged();
                OnPropertyChanged(nameof(IsNotLoading));
            }
        }
    }

    public bool IsNotLoading => !_isLoading;

    public ICommand NavigateToProductCommand { get; }

    public AllProductsPage()
```

```
    {
        InitializeComponent();
        _apiService = new ApiService();

        _allProducts = new ObservableCollection<ProductViewModel>();
        FilteredProducts = new ObservableCollection<ProductViewModel>();
        _currentFilters = new FilterOptions();
        _categoryNames = new Dictionary<int, string>();
        BindingContext = this;

        NavigateToProductCommand = new Command<int>(async (productId) =>
        {
            await
Shell.Current.GoToAsync($"{nameof(ProductDetailPage)}?ProductId={productId}")
;
        });

        LoadData();

        MessagingCenter.Subscribe<FiltersPage, FilterOptions>(this, "ApplyFilters",
(sender, filters) =>
        {
            _currentFilters = filters;
            ApplyFilters();
        });
    }

    private async void LoadData()
    {
        if (IsLoading) return;

        IsLoading = true;
        try
        {
            if (!_categoryNames.Any())
            {
                var categories = await _apiService.GetCategoriesAsync();
                foreach (var category in categories)
                {
                    _categoryNames[category.CategoryId] = category.CategoryName;
                    Console.WriteLine($"Category Loaded: ID={category.CategoryId},
Name={category.CategoryName}");
                }
            }
```

```
            var products = await _apiService.GetProductsAsync();
            _allProducts.Clear();

            foreach (var product in products)
            {
                Console.WriteLine($"Product Loaded: ID={product.ProductId},
Name={product.ProductName}, CategoryId={product.CategoryId}");
            }

            foreach (var product in products)
            {
                var productViewModel = new ProductViewModel
                {
                    ProductId = product.ProductId,
                    Name = product.ProductName,
                    CategoryId = product.CategoryId,
                    CategoryName = _categoryNames.ContainsKey(product.CategoryId) ?
_categoryNames[product.CategoryId] : "Невідома",
                    Price = product.Price,
                    Description = product.Description,
                    Weight = product.Weight,
                    SpicyLevel = product.SpicyLevel,
                    Quantity = product.Quantity,
                    ImageUrl = product.ProductImage
                };
                _allProducts.Add(productViewModel);
            }

            ApplyFilters();
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Error loading data: {ex.Message}");
            await DisplayAlert("Помилка", "Не вдалося завантажити товари.", "OK");
        }
        finally
        {
            IsLoading = false;
        }
    }

    private void ApplyCategoryFilter()
    {
        if (_categoryId > 0)
        {
```

```
            Console.WriteLine($"Applying filter for CategoryId: {_categoryId}");
            _currentFilters.SelectedCategoryIds.Clear();
            _currentFilters.SelectedCategoryIds.Add(_categoryId);
            ApplyFilters();
        }
    }

    private async void OnSearchTextChanged(object sender, TextChangedEventArgs
e)
    {
        var searchText = e.NewTextValue?.Trim().ToLower();
        if (_lastSearchText == searchText) return;
        _lastSearchText = searchText;

        if (_debounceTask != null && !_debounceTask.IsCompleted)
        {
            await _debounceTask;
        }

        _debounceTask = Task.Delay(300).ContinueWith(_ =>
        {
            MainThread.BeginInvokeOnMainThread(() =>
            {
                ApplyFilters();
            });
        });
    }

    private void ApplyFilters()
    {
        FilteredProducts.Clear();

        var filtered = _allProducts.AsEnumerable();

        if (!string.IsNullOrEmpty(_lastSearchText))
        {
            filtered = filtered.Where(p =>
p.Name.ToLower().Contains(_lastSearchText));
        }

        if (_currentFilters.SelectedCategoryIds.Any())
        {
            Console.WriteLine($"Filtering by CategoryIds: {string.Join(", ",
_currentFilters.SelectedCategoryIds)}");
```

```
            filtered = filtered.Where(p =>
_currentFilters.SelectedCategoryIds.Contains(p.CategoryId));
        }

        if (_currentFilters.MinPrice.HasValue)
        {
            filtered = filtered.Where(p => p.Price >= _currentFilters.MinPrice.Value);
        }
        if (_currentFilters.MaxPrice.HasValue)
        {
            filtered = filtered.Where(p => p.Price <= _currentFilters.MaxPrice.Value);
        }

        if (_currentFilters.MinSpicyLevel.HasValue)
        {
            filtered = filtered.Where(p => p.SpicyLevel >=
_currentFilters.MinSpicyLevel.Value);
        }
        if (_currentFilters.MaxSpicyLevel.HasValue)
        {
            filtered = filtered.Where(p => p.SpicyLevel <=
_currentFilters.MaxSpicyLevel.Value);
        }

        foreach (var product in filtered)
        {
            FilteredProducts.Add(product);
        }

        Console.WriteLine($"Filtered Products Count: {FilteredProducts.Count}");
    }

    private async void OnBackTapped(object sender, EventArgs e)
    {
        await Shell.Current.GoToAsync("..");
    }

    private async void OnFilterMenuTapped(object sender, EventArgs e)
    {
        await Shell.Current.GoToAsync(nameof(FiltersPage));
    }

    private async void OnAddToCartTapped(object sender, EventArgs e)
    {
        var frame = sender as Frame;
```

```
        var product = frame?.BindingContext as ProductViewModel;
        if (product != null)
        {
            var cartService = CartService.Instance;
            var cartItem = cartService.CartItems.FirstOrDefault(item =>
item.Product.ProductId == product.ProductId);

            if (cartItem != null)
            {
                int newQuantity = cartItem.Quantity + 1;
                if (newQuantity <= product.Quantity)
                {
                    cartItem.Quantity = newQuantity;
                    await DisplayAlert("Кошик", $"Кількість '{product.Name}' оновлено
до {newQuantity} шт.", "OK");
                }
                else
                {
                    await DisplayAlert("Увага", $"На складі залишилось тільки
{product.Quantity} одиниць товару.", "OK");
                    return;
                }
            }
            else
            {
                if (1 <= product.Quantity)
                {
                    cartService.CartItems.Add(new CartItemViewModel
                    {
                        Product = product,
                        Quantity = 1
                    });
                    await DisplayAlert("Кошик", $"Товар '{product.Name}' додано до
кошика у кількості 1 шт.", "OK");
                }
                else
                {
                    await DisplayAlert("Увага", $"На складі залишилось тільки
{product.Quantity} одиниць товару.", "OK");
                    return;
                }
            }

            MessagingCenter.Send(this, "CartUpdated");
        }
```

```
    }

    protected override void OnAppearing()
    {
        base.OnAppearing();
        ApplyCategoryFilter();
    }

    public new event PropertyChangedEventHandler PropertyChanged;
    protected void OnPropertyChanged([CallerMemberName] string propertyName =
null)
    {
        PropertyChanged?.Invoke(this, new
PropertyChangedEventArgs(propertyName));
    }
}
```

Д.5 – Сторінка кошику

```
using KoreaShop.Models;
using KoreaShop.Services;
using System.ComponentModel;
using System.Runtime.CompilerServices;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;

namespace KoreaShop.Pages
{
    public partial class CartPage : ContentPage, INotifyPropertyChanged
    {
        private readonly CartService _cartService;
        private readonly ApiService _apiService;
        private bool _isLoading;

        public bool IsLoading
        {
            get => _isLoading;
            set
            {
                _isLoading = value;
                OnPropertyChanged();
            }
        }
```

```
    public CartPage()
    {
        InitializeComponent();
        _cartService = CartService.Instance;
        _apiService = new ApiService();
        BindingContext = _cartService;

        MessagingCenter.Subscribe<object>(this, "CartUpdated", (sender) =>
        {
            MainThread.BeginInvokeOnMainThread(() =>
            {
                _cartService.UpdateTotalAmount();
            });
        });

        IsLoading = false;
    }

    protected override void OnAppearing()
    {
        base.OnAppearing();
        _cartService.UpdateTotalAmount();
    }

    private async void OnPlaceOrderClicked(object sender, EventArgs e)
    {
        if (_cartService.CartItems.Count == 0)
        {
            await DisplayAlert("Помилка", "Кошик порожній! Додайте товари перед
оформленням замовлення.", "OK");
            return;
        }

        decimal bonusesToUse = 0;
        if (!string.IsNullOrWhiteSpace(BonusesToUseEntry.Text))
        {
            if (!decimal.TryParse(BonusesToUseEntry.Text, out bonusesToUse) ||
bonusesToUse < 0)
            {
                BonusesErrorLabel.Text = "Введіть додатнє число";
                BonusesErrorLabel.IsVisible = true;
                return;
            }
            if (bonusesToUse > SessionManager.Bonuses)
```

```
            {
                BonusesErrorLabel.Text = "Перевищено доступні бонуси";
                BonusesErrorLabel.IsVisible = true;
                return;
            }
            if (bonusesToUse > _cartService.TotalAmount)
            {
                BonusesErrorLabel.Text = "Бонуси не можуть перевищувати суму чеку";
                BonusesErrorLabel.IsVisible = true;
                return;
            }
            BonusesErrorLabel.IsVisible = false;
        }

        int? customerId = SessionManager.CustomerId;
        if (!customerId.HasValue)
        {
            if (Preferences.ContainsKey("CustomerId"))
            {
                customerId = Preferences.Get("CustomerId", 0);
                SessionManager.CustomerId = customerId;
                SessionManager.Login = Preferences.Get("Login", null);
                SessionManager.Bonuses = decimal.Parse(Preferences.Get("Bonuses", "0"));
                Console.WriteLine($"Loaded from Preferences: CustomerId={customerId}, Login={SessionManager.Login}, Bonuses={SessionManager.Bonuses}");
            }
        }

        Console.WriteLine($"SessionManager.CustomerId: {SessionManager.CustomerId}");
        Console.WriteLine($"Preferences.CustomerId: {Preferences.Get("CustomerId", 0)}");

        if (!customerId.HasValue || customerId == 0)
        {
            await DisplayAlert("Помилка", "Будь ласка, увійдіть у систему для оформлення замовлення.", "OK");
            return;
        }

        var stockErrors = new List<string>();
        foreach (var item in _cartService.CartItems)
```

```
        {
            var product = await _apiService.GetProductAsync(item.Product.ProductId);
            if (product == null || product.Quantity < item.Quantity)
            {
                stockErrors.Add($"Товару '{item.Product.Name}' на складі лише {product?.Quantity ?? 0} одиниць, замовлено {item.Quantity}.");
            }
        }

        if (stockErrors.Any())
        {
            await DisplayAlert("Помилка", "Недостатньо товарів на складі:\n" + string.Join("\n", stockErrors), "OK");
            return;
        }

        decimal finalAmount = bonusesToUse > 0 ? _cartService.TotalAmount - bonusesToUse : _cartService.TotalAmount;

        bool confirm = await DisplayAlert("Підтвердження", $"Ви хочете оформити замовлення на суму ₴{finalAmount}?", "Так", "Ні");
        if (!confirm)
            return;

        IsLoading = true;

        try
        {
            var order = new Order
            {
                CustomerId = customerId.Value,
                OrderDate = DateTime.UtcNow,
                TotalAmount = finalAmount
            };

            var createdOrder = await _apiService.CreateOrderAsync(order);
            if (createdOrder == null || createdOrder.OrderId == 0)
            {
                await DisplayAlert("Помилка", "Не вдалося створити замовлення.", "OK");
                return;
            }

            var errors = new List<string>();
            var successfulDetails = new List<string>();
```

```
            decimal successfulTotal = 0;

            foreach (var item in _cartService.CartItems.ToList())
            {
               try
               {
                  var product = await
_apiService.GetProductAsync(item.Product.ProductId);
                  if (product == null || product.Quantity < item.Quantity)
                  {
                     errors.Add($"Недостатньо товару '{item.Product.Name}' на
складі. Залишок: {product?.Quantity ?? 0}.");
                     continue;
                  }

                  var orderDetail = new OrderDetail
                  {
                     OrderId = createdOrder.OrderId,
                     ProductId = item.Product.ProductId,
                     Quantity = item.Quantity,
                     UnitPrice = item.Product.Price
                  };

                  var createdDetail = await
_apiService.CreateOrderDetailAsync(orderDetail);
                  if (createdDetail == null)
                  {
                     errors.Add($"Не вдалося додати деталі для товару
'{item.Product.Name}': отримана порожня відповідь.");
                     continue;
                  }

                  var newQuantity = product.Quantity - item.Quantity;
                  bool updateSuccess = await
_apiService.UpdateProductQuantityAsync(item.Product.ProductId, newQuantity);
                  if (!updateSuccess)
                  {
                     errors.Add($"Не вдалося оновити залишки для товару
'{item.Product.Name}'.");
                     continue;
                  }

                  successfulDetails.Add($"Додано деталь: {item.Product.Name},
кількість: {item.Quantity}, нові залишки: {newQuantity}");
                  successfulTotal += item.Quantity * item.Product.Price;
```

```
            }
            catch (Exception ex)
            {
                Console.WriteLine($"Error processing item {item.Product.Name}: {ex.Message}");
                errors.Add($"Помилка при обробці товару '{item.Product.Name}': {ex.Message}");
            }
        }

        var customer = await _apiService.GetCustomerByIdAsync(customerId.Value);
        decimal newBonuses = 0;
        decimal bonusesToAdd = 0;

        if (customer == null)
        {
            errors.Add("Не вдалося отримати дані клієнта для оновлення бонусів.");
        }
        else
        {
            if (bonusesToUse > 0)
            {
                newBonuses = customer.Bonuses - bonusesToUse;
            }
            else
            {
                bonusesToAdd = Math.Round(_cartService.TotalAmount * 0.03m);
                newBonuses = customer.Bonuses + bonusesToAdd;
            }

            Console.WriteLine($"Calculated bonuses: Current={customer.Bonuses}, New={newBonuses}");
            await _apiService.UpdateCustomerBonusesAsync(customerId.Value, newBonuses);

            SessionManager.Bonuses = newBonuses;
            Preferences.Set("Bonuses", newBonuses.ToString());
            AvailableBonusesLabel.Text = newBonuses.ToString("F0");
        }

        _cartService.CartItems.Clear();
        _cartService.UpdateTotalAmount();
        BonusesToUseEntry.Text = string.Empty;
```

```
            string finalMessage = $"Замовлення #{createdOrder.OrderId} на суму
₴{createdOrder.TotalAmount} успішно оформлено!";
            if (bonusesToUse > 0)
            {
                finalMessage += $"\nСписано {bonusesToUse} бонусів.";
            }
            else if (bonusesToAdd > 0)
            {
                finalMessage += $"\nНараховано {bonusesToAdd} бонусів.";
            }

            await DisplayAlert("Успішно", finalMessage, "OK");
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Order error: {ex.Message}");
            await DisplayAlert("Помилка", $"Помилка при оформленні замовлення:
{ex.Message}", "OK");
        }
        finally
        {
            IsLoading = false;
        }
    }

    private void OnDeleteTapped(object sender, TappedEventArgs e)
    {
        if (sender is Frame frame && frame.GestureRecognizers.FirstOrDefault() is
TapGestureRecognizer recognizer)
        {
            var item = recognizer.CommandParameter as CartItemViewModel;
            if (item != null)
            {
                _cartService.CartItems.Remove(item);
                _cartService.UpdateTotalAmount();
            }
        }
    }

    private async void OnDecrementTapped(object sender, TappedEventArgs e)
    {
        if (sender is Frame frame && frame.GestureRecognizers.FirstOrDefault() is
TapGestureRecognizer recognizer)
        {
```

```
            var item = recognizer.CommandParameter as CartItemViewModel;
            if (item != null && item.Quantity > 1)
            {
                var product = await _apiService.GetProductAsync(item.Product.ProductId);
                if (product != null && item.Quantity <= product.Quantity)
                {
                    item.Quantity--;
                    _cartService.UpdateTotalAmount();
                }
                else
                {
                    await DisplayAlert("Увага", $"На складі залишилось тільки {product?.Quantity ?? 0} одиниць товару.", "OK");
                }
            }
        }

        private async void OnIncrementTapped(object sender, TappedEventArgs e)
        {
            if (sender is Frame frame && frame.GestureRecognizers.FirstOrDefault() is TapGestureRecognizer recognizer)
            {
                var item = recognizer.CommandParameter as CartItemViewModel;
                if (item != null)
                {
                    var product = await _apiService.GetProductAsync(item.Product.ProductId);
                    if (product != null && item.Quantity < product.Quantity)
                    {
                        item.Quantity++;
                        _cartService.UpdateTotalAmount();
                    }
                    else
                    {
                        await DisplayAlert("Увага", $"На складі залишилось тільки {product?.Quantity ?? 0} одиниць товару.", "OK");
                    }
                }
            }
        }

        public event PropertyChangedEventHandler PropertyChanged;
```

```
        protected void OnPropertyChanged([CallerMemberName] string propertyName
= null)
        {
            PropertyChanged?.Invoke(this,                                    new
PropertyChangedEventArgs(propertyName));
        }
    }
}
```

Д.6 – Сторінка профілю користувача

```
using System.ComponentModel;
using System.Collections.ObjectModel;
using Microsoft.Maui.Controls;
using KoreaShop.Services;
using KoreaShop.Models;
using System.Threading.Tasks;
using System.Linq;

namespace KoreaShop.Pages;

public partial class ProfilePage : ContentPage, INotifyPropertyChanged
{
    private string _greeting;
    private string _login;
    private string _cardCode;
    private string _bonuses;
    private string _cardCodeDisplay;
    private int _orderItemsCount;
    private ObservableCollection<OrderViewModel> _orderItems;
    private readonly ApiService _apiService;

    public string Greeting
    {
        get => _greeting;
        set
        {
            _greeting = value;
            OnPropertyChanged(nameof(Greeting));
        }
    }

    public string Login
    {
        get => _login;
```

```
    set
    {
        _login = value;
        OnPropertyChanged(nameof(Login));
    }
}

public string CardCode
{
    get => _cardCode;
    set
    {
        _cardCode = value;
        CardCodeDisplay = $"Бонусна картка: {value}";
        OnPropertyChanged(nameof(CardCode));
    }
}

public string Bonuses
{
    get => _bonuses;
    set
    {
        _bonuses = value;
        OnPropertyChanged(nameof(Bonuses));
    }
}

public string CardCodeDisplay
{
    get => _cardCodeDisplay;
    set
    {
        _cardCodeDisplay = value;
        OnPropertyChanged(nameof(CardCodeDisplay));
    }
}

public int OrderItemsCount
{
    get => _orderItemsCount;
    set
    {
        _orderItemsCount = value;
        OnPropertyChanged(nameof(OrderItemsCount));
```

```
    }
  }

  public ObservableCollection<OrderViewModel> OrderItems
  {
    get => _orderItems;
    set
    {
      _orderItems = value;
      OnPropertyChanged(nameof(OrderItems));
    }
  }

  public ProfilePage()
  {
    InitializeComponent();
    _apiService = new ApiService();
    OrderItems = new ObservableCollection<OrderViewModel>();
    BindingContext = this;
  }

  protected override async void OnAppearing()
  {
    base.OnAppearing();
    System.Diagnostics.Debug.WriteLine("ProfilePage: OnAppearing called");
    await LoadUserDataAsync();
  }

  private async Task LoadUserDataAsync()
  {
    try
    {
      System.Diagnostics.Debug.WriteLine("ProfilePage: Loading user data");
      int? customerId = SessionManager.CustomerId;
      if (!customerId.HasValue && Preferences.ContainsKey("CustomerId"))
      {
        customerId = Preferences.Get("CustomerId", 0);
        SessionManager.CustomerId = customerId;
        SessionManager.Login = Preferences.Get("Login", null);
        SessionManager.Bonuses = decimal.Parse(Preferences.Get("Bonuses",
"0"));
        System.Diagnostics.Debug.WriteLine($"ProfilePage: CustomerId loaded
from Preferences: {customerId}");
      }
```

```
        if (!customerId.HasValue || customerId == 0)
        {
            System.Diagnostics.Debug.WriteLine("ProfilePage: No valid CustomerId
found");
            await DisplayAlert("Помилка", "Будь ласка, увійдіть у систему.",
"OK");
            return;
        }

        var customer = await
_apiService.GetCustomerByIdAsync(customerId.Value);
        if (customer == null)
        {
            System.Diagnostics.Debug.WriteLine("ProfilePage: Failed to retrieve
customer data");
            await DisplayAlert("Помилка", "Не вдалося отримати дані
користувача.", "OK");
            return;
        }

        Greeting = $"Привіт, {customer.FullName}!";
        Login = customer.Login;
        CardCode = string.IsNullOrEmpty(customer.CardCode) ? "Невідомий код" :
customer.CardCode;
        Bonuses = customer.Bonuses.ToString("F0");
        System.Diagnostics.Debug.WriteLine($"ProfilePage: Customer data loaded -
FullName: {customer.FullName}, Login: {customer.Login}");

        Preferences.Set("FullName", customer.FullName);
        Preferences.Set("Login", customer.Login);
        Preferences.Set("CardCode", customer.CardCode);
        Preferences.Set("Bonuses", customer.Bonuses.ToString());

        var allOrders = await _apiService.GetOrdersAsync();
        var customerOrders = allOrders.Where(o => o.CustomerId ==
customerId.Value).OrderByDescending(o => o.OrderDate).ToList();
        OrderItemsCount = customerOrders.Count();
        OrderItems.Clear();
        System.Diagnostics.Debug.WriteLine($"ProfilePage: Loaded
{customerOrders.Count} orders for CustomerId: {customerId}");

        var allOrderDetails = await _apiService.GetOrderDetailsAsync();
        System.Diagnostics.Debug.WriteLine($"ProfilePage: Loaded
{allOrderDetails.Count} order details");
```

```
            foreach (var order in customerOrders)
            {
                var orderDetails = allOrderDetails.Where(d => d.OrderId ==
order.OrderId).ToList();
                decimal calculatedTotal = orderDetails.Sum(detail => detail.Quantity *
detail.UnitPrice);

                string bonusesInfo;
                if (calculatedTotal == order.TotalAmount)
                {
                    var bonusesEarned = Math.Round(order.TotalAmount * 0.03m);
                    bonusesInfo = $"Нараховано бонусів: {bonusesEarned}";
                }
                else if (calculatedTotal > order.TotalAmount)
                {
                    var bonusesUsed = calculatedTotal - order.TotalAmount;
                    bonusesInfo = $"Списано бонусів: {bonusesUsed}";
                }
                else
                {
                    bonusesInfo = "Помилка в обрахунку бонусів";
                }

                OrderItems.Add(new OrderViewModel
                {
                    OrderId = order.OrderId,
                    OrderDate = order.OrderDate,
                    TotalAmount = order.TotalAmount,
                    BonusesInfo = bonusesInfo
                });
                System.Diagnostics.Debug.WriteLine($"ProfilePage: Added order -
OrderId: {order.OrderId}, TotalAmount: {order.TotalAmount}");
            }
        }
        catch (Exception ex)
        {
            System.Diagnostics.Debug.WriteLine($"ProfilePage: Error loading user data:
{ex.Message}");
            await DisplayAlert("Помилка", $"Помилка при завантаженні даних:
{ex.Message}", "OK");
        }
    }

    private async void OnOrderTapped(object sender, TappedEventArgs e)
    {
```

```
        try
        {
            System.Diagnostics.Debug.WriteLine("ProfilePage: OnOrderTapped
triggered");
            var order = e.Parameter as OrderViewModel;
            if (order == null)
            {
                System.Diagnostics.Debug.WriteLine("ProfilePage: OrderViewModel is
null in OnOrderTapped");
                await DisplayAlert("Помилка", "Не вдалося отримати дані
замовлення.", "OK");
                return;
            }

            System.Diagnostics.Debug.WriteLine($"ProfilePage: Order tapped - OrderId:
{order.OrderId}, TotalAmount: {order.TotalAmount}");
            System.Diagnostics.Debug.WriteLine("ProfilePage: Navigating to
OrderDetailsPage");
            await Navigation.PushAsync(new OrderDetailsPage(order.OrderId,
order.TotalAmount));
            System.Diagnostics.Debug.WriteLine("ProfilePage: Navigation to
OrderDetailsPage completed");
        }
        catch (Exception ex)
        {
            System.Diagnostics.Debug.WriteLine($"ProfilePage: Error in
OnOrderTapped: {ex.Message}");
            await DisplayAlert("Помилка", $"Помилка при переході до деталей
замовлення: {ex.Message}", "OK");
        }
    }

    private async void OnLogoutClicked(object sender, EventArgs e)
    {
        try
        {
            System.Diagnostics.Debug.WriteLine("ProfilePage: Logout button clicked");
            bool confirm = await DisplayAlert("Підтвердження", "Ви впевнені, що
хочете вийти?", "Так", "Ні");
            if (!confirm)
            {
                System.Diagnostics.Debug.WriteLine("ProfilePage: Logout cancelled by
user");
                return;
            }
```

```
            SessionManager.Clear();
            Preferences.Remove("CustomerId");
            Preferences.Remove("FullName");
            Preferences.Remove("Login");
            Preferences.Remove("Bonuses");
            Preferences.Remove("CardCode");
            Application.Current.MainPage = new NavigationPage(new LoginPage());
            System.Diagnostics.Debug.WriteLine("ProfilePage: Successfully logged out
and navigated to LoginPage");
        }
        catch (Exception ex)
        {
            System.Diagnostics.Debug.WriteLine($"ProfilePage: Error during logout:
{ex.Message}");
            await DisplayAlert("Помилка", $"Помилка при виході: {ex.Message}",
"OK");
        }
    }

    public new event PropertyChangedEventHandler PropertyChanged;

    protected void OnPropertyChanged(string propertyName)
    {
        PropertyChanged?.Invoke(this, new
PropertyChangedEventArgs(propertyName));
    }
}

public class OrderViewModel
{
    public int OrderId { get; set; }
    public DateTime OrderDate { get; set; }
    public decimal TotalAmount { get; set; }
    public string BonusesInfo { get; set; }
}
```