МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Чорноморський національний університет
імені Петра Могили
Факультет комп'ютерних наук
Кафедра інтелектуальних інформаційних систем

ДОПУЩЕНО ДО ЗАХИСТУ
Завідувач кафедри інтелектуальних інформаційних систем, д-р техн. наук, проф.
_____Ю. П. Кондратенко
«____»_____2022 р.

БАКАЛАВРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

МОБІЛЬНИЙ ЗАСТОСУНОК ДЛЯ ІНТЕРНЕТ-МАГАЗИНУ З ПРОДАЖУ ЕЛЕКТРОНІКИ

Спеціальність 122 «Комп'ютерні наук»

122 – БКР – 402.21810221

Виконав студент 4-го курсу, групи 402
_____ І. С. Раздобудько
«__» червня 2022 р.

Керівник: ст.. викладач
_____В. В. Кошовий
«__» червня 2022 р.

Миколаїв – 2022

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Чорноморський національний університет
імені Петра Могили
Факультет комп'ютерних наук
Кафедра інтелектуальних інформаційних систем

Рівень вищої освіти **бакалавр**
Спеціальність **122 «Комп'ютерні науки»**
(шифр і назва)
Галузь знань **12 «Інформаційні технології»**
(шифр і назва)

ЗАТВЕРДЖУЮ

Завідувач кафедри інтелектуальних інформаційних систем, д-р техн. наук, проф.
_____ Ю.П. Кондратенко
«____» _____ 2022 р.

ЗАВДАННЯ
на виконання кваліфікаційної роботи

Видано студенту групи 402 факультету комп'ютерних наук

Раздобудько Івану Сергійовичу

1. Тема кваліфікаційної роботи

Мобільний застосунок для інтернет-магазину з продажу електроніки

Керівник роботи : старший викладач Віталій Володимирович Кошовий

Затв. наказом Ректора ЧНУ ім. Петра Могили від «_07_» __12__ 2021р. №_318_

2. Строк представлення кваліфікаційної роботи студентом «__» _____ 2022р.

3. Вхідні(початкові) дані роботи: мобільний застосунок, середовище для розробки програмного забезпечення, технології розробки.

Очікуваний результат роботи: мобільний застосунок інтернет-магазину з продажу електроніки.

4. Перелік питань, що підлягають розробці (зміст пояснювальної записки):

- аналіз ринку мобільних застосунків;

- вибір програмного забезпечення для створення мобільного застосунку та моделювання систему програмного забезпечення;
- програмна реалізація мобільного застосунку інтернет-магазину.

5. Перелік графічних матеріалів: сторінок – 98, таблиць – 8, рисунків – 56, посилань – 26, презентація

6. Завдання до спеціальної частини

Охорона праці на робочому місці у кабінеті у ЧНУ ім. Петра Могили

7. Консультанти:

Розділ	Прізвище, ініціали та посада консультанта	Підпис
Спеціальна частина з охороною праці	Макарова О.В., старший викладач	

Керівник роботи : старший викладач Кошовий В. В.

(підпис)

Завдання прийнято до виконання Раздобудько Іван Сергійович

(прізвище, ім’я, по батькові студента)

(підпис)

Дата видачі завдання «01» грудня 2021 р.

КАЛЕНДАРНИЙ ПЛАН
на виконання кваліфікаційної роботи

Тема: «Мобільний застосунок для інтернет-магазину з продажу електроніки»

№	Найменування роботи	Початок	Закінчення	Примітки
1	Подання заяви на затвердження теми та керівників БКР	15.10.2021	15.10.2021	виконано
2	Отримання завдання на виконання БКР	16.10.2021	16.10.2021	виконано
3	Складання календарного плану роботи на весь період виконання БКР	19.10.2021	19.10.2021	виконано
4	Отримання завдання на переддипломну практику	06.04.2022	06.04.2022	виконано
5	Проходження переддипломної практики, збір та аналіз матеріалів до БКР	20.04.2022	09.05.2022	виконано
6	Розробка звіту з переддипломної практики	10.05.2022	12.05.2022	виконано
7	Виконання БКР: аналіз ринку мобільних застосунків, вибір підходів для створення застосунку, створення застосунку, тестування	13.05.2022	27.05.2022	виконано
8	Попередній захист БКР на засіданні комісії кафедри	31.05.2022	31.05.2022	виконано
9	Доробка та остаточно оформлення БКР	04.05.2022	08.05.2022	виконано
10	Подання БКР рецензенту	10.05.2022	10.05.2022	виконано
11	Подання БКР, її електронної копії та інших документів (відгуку, рецензії) до захисту	21.06.2022	25.06.2022	
12	Захист БКР перед екзаменаційною комісією (ЕК)	27.06.2022	27.06.2022	

Розробив студент __Раздобудько І. С._____ _____
(прізвище, ім'я, по батькові студента) (*підпис*)

Керівник роботи _ __ст. викладач Кошовий В. В._____ _____
(*посада, прізвище, ім'я, по батькові)* *(підпис)*

«_12_» _____12_____ 2021 р.

АНОТАЦІЯ

бакалаврської кваліфікаційної роботи студента групи 402 ЧНУ ім. Петра Могили

Раздобудько Івана Сергійовича

Тема: «Мобільний застосунок для інтернет-магазину з продажу електроніки»

Керівник: старший викладач Віталій Володимирович Кошовий.

Об’єкт дослідження – процес створення мобільного застосунку інтернет-магазину з продажу електроніки.

Предмет дослідження – засоби для створення мобільного застосунку інтернет-магазину з продажу електроніки.

Мета – метою роботи є розробка та впровадження мобільного клієнта інтернет-магазину, що дозволяє здійснювати покупки з доставкою на пошту за допомогою смартфонів із системою Android.

У першому розділі розглядаються та здійснюється аналізу ринку мобільних застосунків та вимог до них.

У другому розділі – процес моделювання та проектування мобільного застосунку.

У третьому розділі представлено опис процесу розробки мобільного застосунку.

В останньому розділі було розглянуто норми та заходи з охорони праці й техніки безпеки під час праці з екранними пристроями

В результаті виконаної роботи було зроблено висновки щодо актуальності роботи та використаних під час створення Застосунку технологій.

Сторінок – 96, таблиць – 8, рисунків – 56, посилань – 26.

Ключові слова: клієнт інтернет-магазину, Android, Kotlin, мобільний застосунок.

ABSTRACT

for bachelor's qualification work of a student of 402 group at Petro Mohyla

Black Sea National University

Razdobudko Ivan Sergeevich

Topic: "Mobile application for an online store selling electronics "

Supervisor: senior lecturer Vitaliy Volodymyrovych Koshovy.

The object of research is the process of creating a mobile application of an online store selling electronics.

The subject of research - tools for creating a mobile application of an online store selling electronics.

Purpose - the purpose of the work is to develop and implement a mobile online store, which allows you to make purchases with delivery by mail using smartphones with Android.

The first section examines and analyzes the market for mobile application requirements and requirements

In the second section - the process of modeling and designing a mobile application.

The third section describes the process of developing a mobile application.

In the last section the norms and measures on labor protection and safety at work with screen devices were considered.

As a result of the work performed, conclusions were made on the relevance of the work and the relevance of the technologies used during the creation of the application.

Pages - 96, tables - 8, figures - 56, links - 26.

Keywords: online store client, Android, Kotlin, mobile application.

Зміст

Пояснювальна записка

до кваліфікаційної роботи

на тему:

«МОБІЛЬНИЙ ЗАСТОСУНОК ДЛЯ ІНТЕРНЕТ-МАГАЗИНУ З ПРОДАЖУ ЕЛЕКТРОНІКИ»

Спеціальність 122 «Комп'ютерні науки»

122 – БКР – 402.21810221

Виконав студент 4-го курсу, групи 402

_____*І. С. Раздобудько*

(підпис, ініціали та прізвище)

«___» _____ 202_ р.

Керівник: _____*ст. викладач*

(наук. ступінь, вчене звання)

_____*В. В. Кошовий*

(підпис, ініціали та прізвище)

«___» _____ 202_ р.

Миколаїв – 2022

Вступ

Характерною ознакою сучасного людства є виробництво, збирання та розподіл. Наше століття можна сміливо назвати віком комп'ютерів і смартфонів та Інтернету. Коли в минулому столітті був винайдений телефон, ніхто не очікував, що це технологічне диво зміниться настільки суттєво. Досягнення в галузі фізики, електроніки та інформатики привели до створення мобільного телефону – сьогодні незамінного засобу зв'язку, який завжди під рукою. Попит на телефони постійно зростає. Сучасні технології дозволяють комфортно працювати та економити час. Мобільний телефон – це не лише засіб міжособистісного спілкування, а й передача різноманітних даних за допомогою спеціальних функцій, можна навіть сказати, що мобільний телефон – це міні-комп'ютер. У зв'язку з попитом на технології та зв'язок виникають організації, які дозволяють здійснювати купівлю-продаж товарів усіх видів. З кожним днем їх стає все більше.

Інтернет як засіб інформації існує вже давно. Для більшості людей, які користуються Інтернетом, це спосіб отримати певну інформацію. Саме так використовуються різні браузери, щоб полегшити його пошук, або вони використовуються людьми для спілкування один з одним. Досить часто Інтернет використовується в бізнесі для продажу товарів і послуг онлайн. Інтернет-магазин дозволяє розміщувати велику кількість інформації, пов'язаної з розпродажем, наприклад, пропозиції товарів, ціни, описи, просування рекламних фільмів, інформацію про акції та знижки, купони, що дозволяють клієнту робити покупки в дуже зручний спосіб. Інтернет-магазин дозволяє швидко оновлювати асортимент, дозволяє контролювати робочі процеси, наприклад, автоматичне оновлення прайс-листів. Крім того, це може бути продовженням традиційного бізнесу або цілком самостійною структурою.

Об’єкт дослідження – процес створення мобільного застосунку інтернет-магазину з продажу електроніки.

Предмет дослідження – засоби для створення мобільного застосунку інтернет-магазину з продажу електроніки.

Мета – метою роботи є розробка та впровадження мобільного клієнта інтернет-магазину, що дозволяє здійснювати покупки з доставкою на пошту за допомогою смартфонів із системою Android.

Для виконання мети треба виконати наступні дії:

• аналіз ринку операційних систем для смартфонів;

• аналіз ринку інтернет-магазинів;

• порівняйте програми, які дозволяють робити покупки в Інтернеті;

• дослідження можливостей застосунків для покупок в Інтернеті;

• створення функціональних / нефункціональних вимог до програми;

• створити базу даних;

• реалізація програми відповідно до проекту.

У ході виконання роботи була зроблена спроба створити мобільний застосунок. Основним припущенням створеної програми був мінімалізм форми та простота використання, що дозволяє безперебійно здійснювати покупки для користувачів усіх вікових груп. Спочатку була створена логіка навігації, заснована на переміщенні між вправами за допомогою тематичного меню в нижній частині екрана. Наступним кроком було підключення програми до бази даних, щоб користувач міг увійти, зареєструватися та вийти. Базу даних товарів доповнюють співробітники магазину - користувачі в ролі адміністратора. Адміністратор може додати товар до магазину з описом, назвою та ціною. Користувачі створюють ієрархію від найнижчого до найвищого, тобто від користувача, який не ввійшов у систему до адміністратора. Для аутентифікації та авторизації користувачів використовувався механізм Firebase.

1. АНАЛІЗ ПОТРЕБ РИНКУ

Мобільні програми полегшують життя, але не всі користуються ними. За даними Forrester, на кінець 2016 року лише 46% населення планети мають при собі мобільний телефон, що залишає багато можливостей для розробки застосунків у цьому напрямку.

Очікується, що наприкінці 2022 році кількість мобільних телефонів досягне близько 3,8 мільярдів, або близько 66% населення світу.

На ринку є багато застосунків та інтернет-магазинів. Сьогодні ми маємо дві найбільші платформи щодо операційних систем: iOS та Android (Apple та Google), які мають два найбільших магазину.

1.1. Поточний стан та визначення потреб

У 2016 році Gartner представив таку статистику: 87,8% проданих мобільних телефонів склали Android, а iOS – 11,5%.

Таблиця 1.1 – Статистика телефонів, проданих у 2016 році

Постачальник	**3Q16 Units**	**3Q16 Market Share (%)**	**3Q15 Units**	**3Q15 Market Share (%)**
Samsung	71,733.5	19.2	83,586.7	23.6
Apple	43,000.7	11.5	46,062.0	13.0
Huawei	32,489.5	8.7	27,412.7	7.7
Oppo	24,936.6	6.7	11,868.6	3.4
BBK Communication Equipment	19,878.9	5.3	10,437.4	2.9
Інші	181,253.3	48.6	174,812.8	49.4
Всього	**373,292.5**	**100.0**	**354,180.2**	**100.0**

У свою чергу, зміни на ринку мобільних телефонів у 2021-2022 рр. показано на рисунку 1.1.

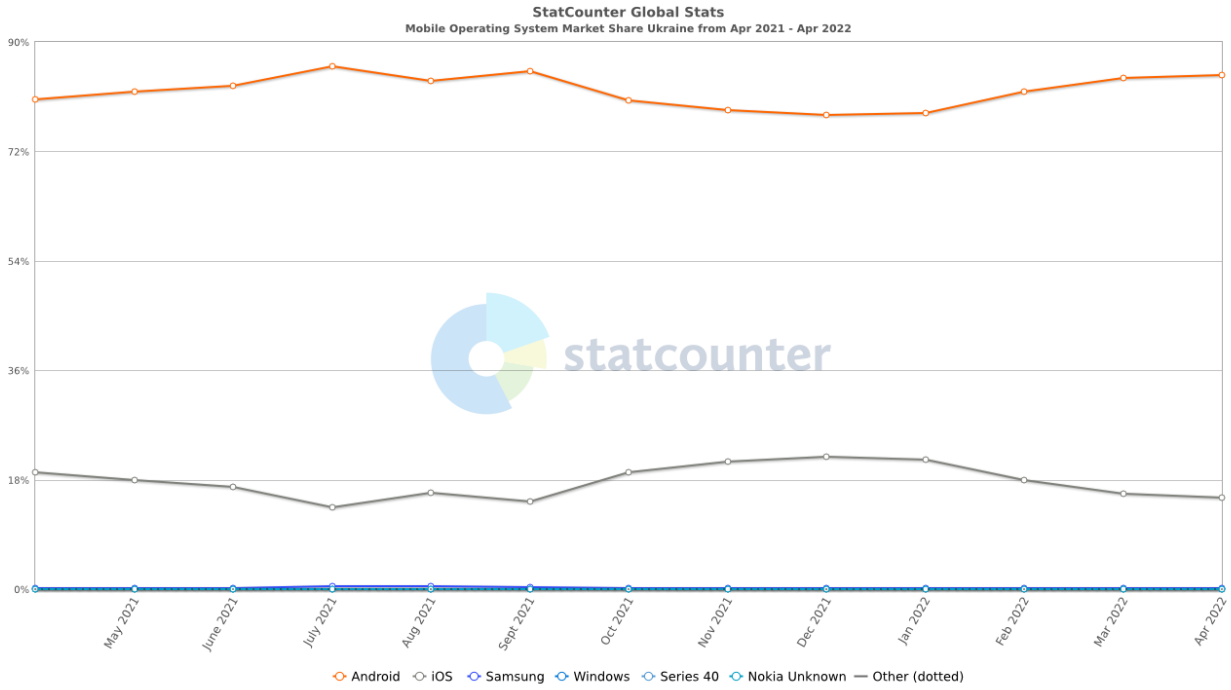

Рисунок 1.1 – Статистика проданих телефонів в Україні 2021 - 2022 рр.

Ми бачимо, що в Україні наприкінці 2021 року продано 78% смартфонів на Android і 22% для iOS.

Це не означає, що програми для Android приносять більше доходу на місяць, ніж програми для iOS. Forbes підрахував, що Застосунок iOS заробляє в середньому 4000 доларів на місяць; Друге місце посів Android, за який один Застосунок заробляв в середньому 1125 доларів на місяць:

	Google	Apple	Microsoft
Average revenue per app	$1125	$4000	$625
Average revenue per developer	$6,000	$21,276	$2,222

Рисунок 1.2 – Різниця між доходами APPLE і GOOGLE

У 2022 році користувачі витратить на мобільні додатки приблизно 133 мільярди доларів, що на 20% більше, ніж у минулому році. В усьому світі

споживчі витрати на App Store становитимуть 85,1 мільярда доларів, що на 17,7 відсотка збільшиться в порівнянні з аналогічним періодом минулого року з 72,3 мільярда доларів у 2020 році.

Безпрецедентне збільшення онлайн-релізів і пов'язане з цим використання даних, особливо в країнах, що розвиваються, таких як Індія, Китай та Бразилія, є основною рушійною силою зростання ринку.

Дивлячись на ринок інтернет-магазинів у США у 2020 році, можна помітити, що роздрібні продажі мобільних телефонів і планшетів становили 44% від загального обсягу роздрібних продажів.

Сегмент ігрових застосунків на ринку мобільних пристроїв мав найвищий дохід, який склав понад 40% усіх доходів у 2019 році, за ним слідує музика. Крім того, очікується, що цей сегмент збереже своє домінування через збільшення кількості ігор та їх зростання популярності в таких країнах, як Китай та Індія. У 2019 році операційна система Android була відповідальною за значну кількість завантажень, тоді як iOS приносила більший дохід.

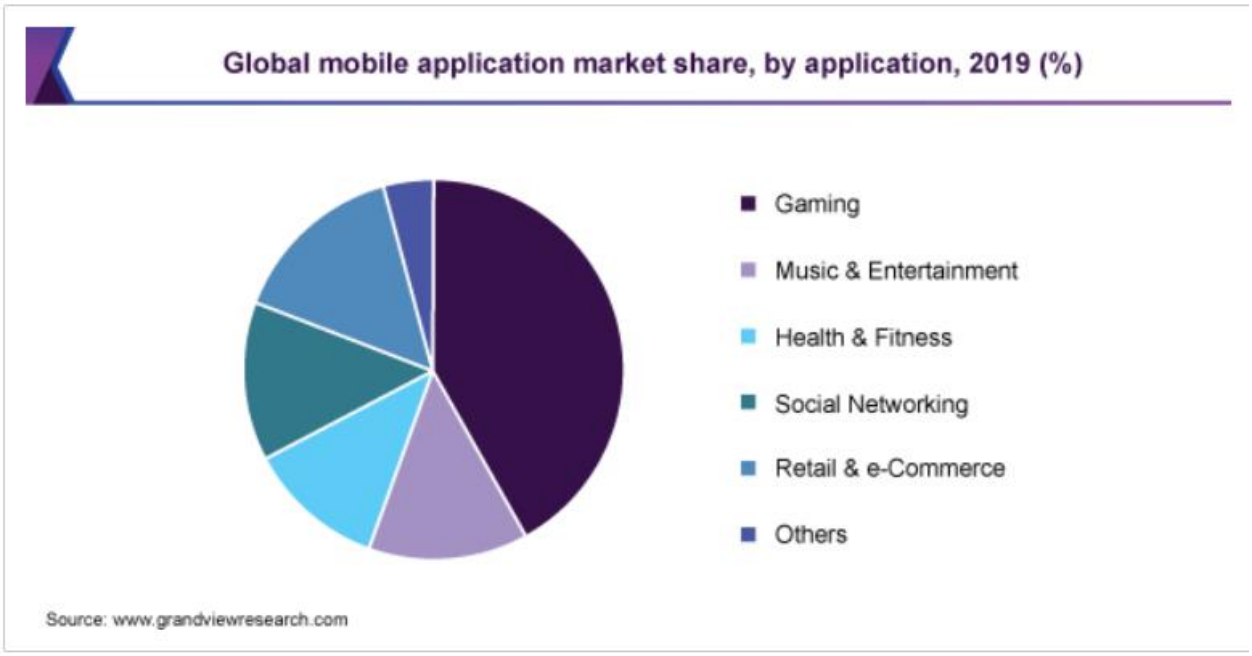

Рисунок 1.3 – Частка світового ринку мобільних застосунків.

За останні кілька років дуже популярними стали інтернет-магазини та програми, які дозволяють використовувати їх через смартфон або комп’ютер. Пандемія COVID-19 призвела до збільшення кількості інтернет-магазинів, найпопулярнішими з них є: Amazon, eBay, Etsy, InspireUplift .

Щоб краще зрозуміти ринок мобільних застосунків, розглянемо найпопулярніші програми для Android та iOS.

Популярні програми для смартфонів Android:

Найкращі безкоштовні програми:

1. Google Pay: безпечний і корисний спосіб керувати грошима
2. TikTok
3. IRS2Go

Найвигідніші програми:

1. Google One
2. Disney+
3. Twitch: пряма трансляція багатокористувацьких ігор та кіберспорту

Популярні програми для смартфонів з iOS:

Найкращі безкоштовні програми:

1. Хмарні зустрічі ZOOM
2. TikTok
3. Disney+
4. YouTube

1.2. Огляд торгових систем

У цьому розділі описано два застосунки: Rozetka та Amazon, а також параметри входу та виходу користувача для кожної програми. На сторінках програми представлено, які види діяльності може виконувати споживач. Rozetka обрали, бо це один із найбільших магазинів в Україні та має великий функціонал, а Amazon вирізняється рекордною кількістю онлайн-товарів у всьому світі.

1.2.1. Застосунок Rozetka

Інтернет-магазин Rozetka – застосунок, що дозволяє купувати електронне обладнання. Ви можете шукати різноманітні товари та акції, доступні в цьому магазині.

Головна:

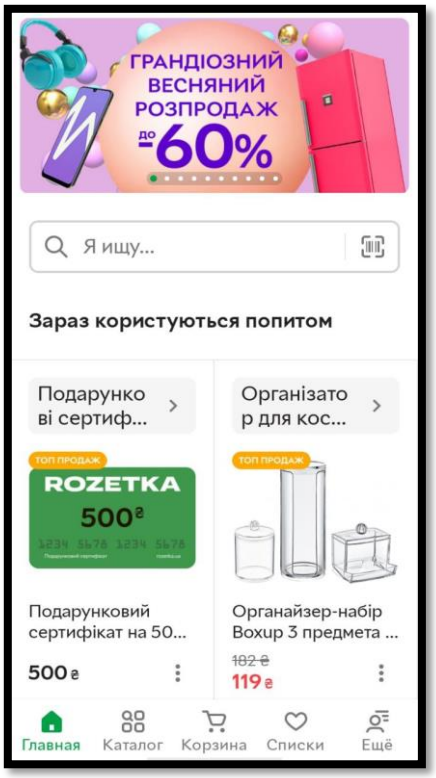

Рисунок 1.4 – Головна сторінка

Навіть якщо користувач не ввійшов у систему, він може переглядати сторінки. Ви можете шукати продукти, ввівши їх назву та використавши штрих-код. Споживач має можливість переглянути найпопулярніші товари чи акції.

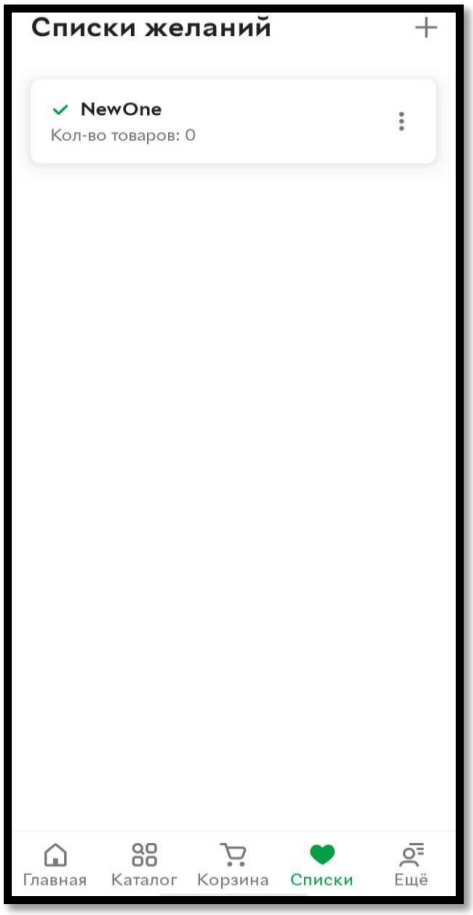

Рисунок 1.5 – Сторінка вибраного товару

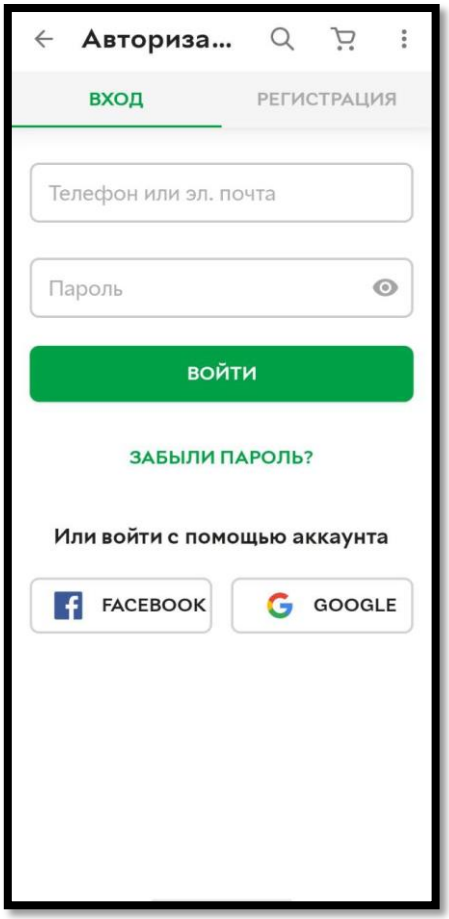

Рисунок 1.6 – Реєстрація

На рисунку 1.5 показана сторінка для створення списків покупок. Однак створити новий не можна, оскільки користувач не ввійшов у систему. Після натискання знака «+» з’явиться сторінка, показана на рисунку 1.6. Ви можете помітити, що є й інші способи входу, наприклад, за допомогою облікового запису Facebook або Google, номера телефону або електронної пошти.

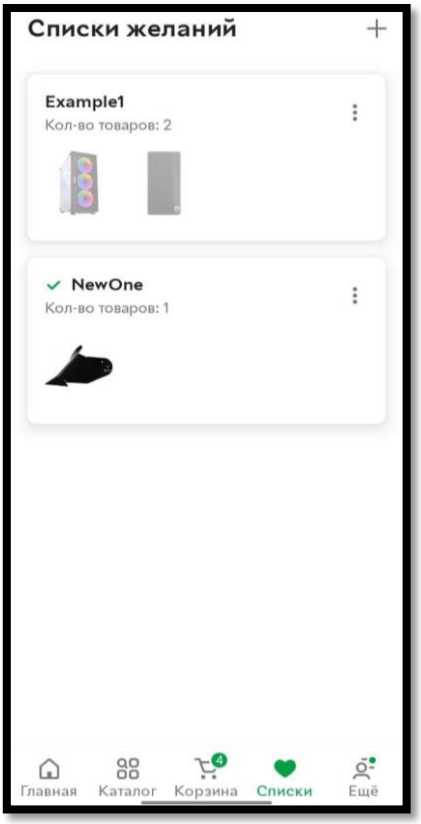

Рисунок 1.7 – Після реєстрації в застосунку

На рисунку 1.7 показано, що під час входу користувач може створювати нові списки, видаляти існуючі, перейменовувати їх, а також видаляти або додавати продукти до списку. Список містить зображення доданих до нього продуктів і кількість цих позицій.

А також, існують додаткові функції, такі як: мої замовлення, сповіщення, переглянуті продукти, дії, знижки та обслуговування клієнтів.

Таблиця 1.2 – Переваги та недоліки застосунку «Rozetka».

Недоліки	Переваги
Неможливо вибрати потрібну мову.	Можлива доставка у відповідний магазин або на пошту.

Закінчення таблиці 1.2.

Акційний продукт не завжди буде в найближчому магазині обраної мережі.	Пошук вибраного товару.
	Швидкість дії (за декілька кліків ви можете перевірити всі акції на товар, що цікавить.)
	Можливість перегляду без входу в застосунок
	Можливість створення кошика для покупок (з оновлення в січні 2022 року)

1.2.2. Застосунок Amazon

Amazon – американська торгова компанія, створена акціонерна компанія у 1994 році в Сіетлі. Він займається електронною комерцією B2C і керує найбільшим у світі інтернет-магазином.

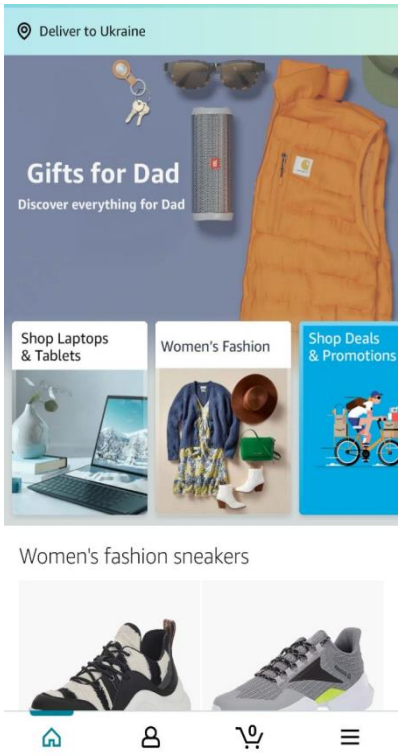

Рисунок 1.8. Головна сторінка

Користувач може зайти на домашню сторінку трьома способами: увійшовши, зареєструвавшись і без входу, що перешкоджає певній діяльності. На домашній сторінці споживач може шукати продукти, переглядати найкращі пропозиції та отримувати інформацію про доставку продукції.

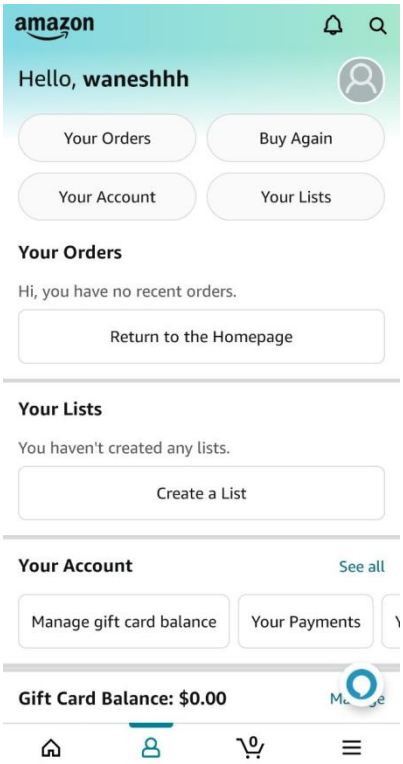

Рисунок 1.9 – Сторінка даних

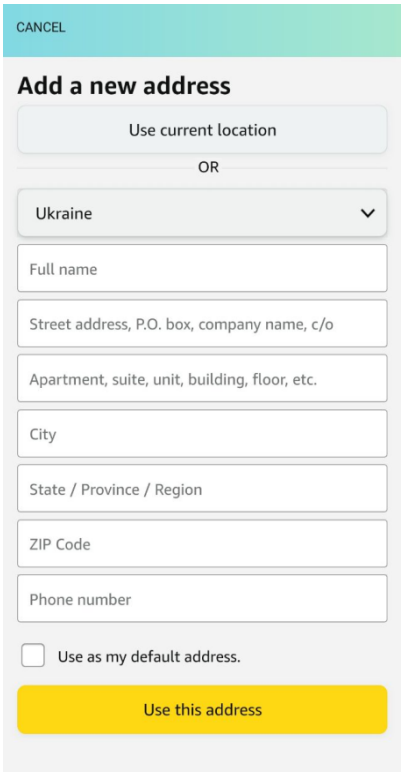

Рисунок 1.10 – Сторінка замовлення

На рисунку 1.9 показана сторінка з даними користувача. На цій сторінці є список покупок, замовлень і є можливість купувати продукти знову, перевірка даних, введених раніше.

На рисунку 1.10 показана сторінка замовлення. Користувач повинен ввести свої персональні дані: ім'я, прізвище, номер телефону, вулицю, номер будинку, поштовий індекс і місто, або користувач може скористатися кнопкою «використати поточне місцезнаходження». Людина, яка використовує застосунок, має можливість додати доставку інструкції, хоча це не є обов’язковим.

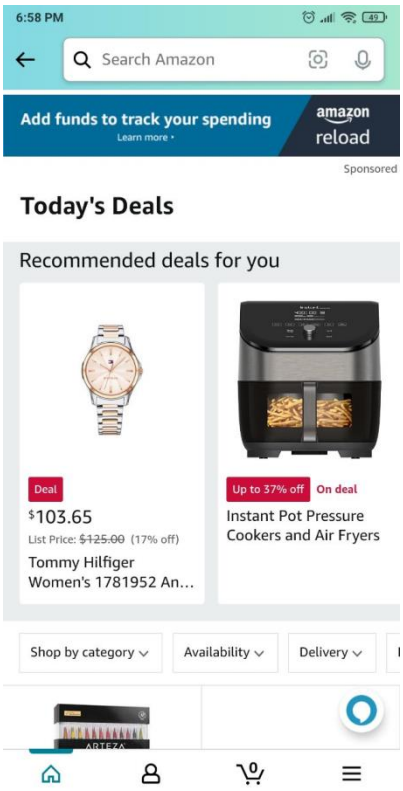

Рисунок 1.11 – Сторінка з пропозиціями

На рисунку 1.11 користувач має можливість вибрати пропозицію, можна вибрати категорії та наявність.

Таблиця 1.3 – Переваги та недоліки застосунку «Amazon»

Недоліки	Переваги
Мала кількість можливих мов на вибір.	Можливість перегляду попередніх покупок.
Доставка в магазини не здійснюється, тільки на пошту чи додому.	У застосунку є Amazon Prime, який має багато різноманітних пропозицій.
Зміна країни/мови викликає повторний запуск програми.	Швидкість дії (за декілька кліків можна перевірити всі акції на цікавий товар)
	Можливість перегляду без входу в застосунок.
	Можливість перегляду без входу в застосунок.

Висновки до розділу 1

У даному розділі було розглянуто ринок мобільних застосунків. Проаналізовано доцільність створення застосунку інтернет-магазину. Для майбутнього моделювання були проаналізовані основні можливості та характеристики кожного елементу, на основі двох прикладів популярних застосунків.

2. МОДЕЛЮВАННЯ ТА ПРОЄКТУВАННЯ МОБІЛЬНОГО ЗАСТОСУНКУ

2.1. Функціональні вимоги системи

1. Адміністратори:
    1.1. Додавання, видалення та редагування продуктів
    1.2. Управління замовленнями.
2. Зареєстрований користувач:
    2.1. Підтримка користувачів:
        2.1.1. Аутентифікація
            2.1.1.1. Увійти
            2.1.1.2. Реєстрація
                2.1.1.2.1. Надання користувачеві прав звичайного користувача
        2.1.2. Редагування облікового запису
            2.1.2.1. Зміна персональних даних (логін або ім'я)
            2.1.2.2. Зміна пароля
            2.1.2.3. Змінити електронну пошту
            2.1.2.4. Змінити аватар
            2.1.2.5. Змінити мову
            2.1.2.6. Змінити тему
        2.1.3. Список рекомендованих продуктів для користувача
            2.1.3.1. Автоматичне створення списку
            2.1.3.2. Автоматичне оновлення списку на основі переглянутих товарів і покупок
    2.2. Виплати:
        2.2.1. Гаманець
            2.2.1.1. Поповнення гаманця

2.2.2. Кредитна карта

2.2.2.1. Додавання картки

2.2.2.2. Видалення карти

2.2.2.3. Перевірка даних картки

2.2.2.4. Перевірка наявності коштів на рахунку

2.2.3. Оплата товару

2.2.4. Вибір оплати

2.2.4.1. Переказ на банківський рахунок (передоплата)

2.2.4.2. Кредитна / дебетова картка

2.3. Продукти:

2.3.1. Відображення каталогу продукції

2.3.2. Пошук продуктів

2.3.2.1. Знайдіть певний продукт, вказавши його назву

2.3.2.2. Вказівка детальних вимог до даного товару і показ продуктів, які відповідають вашим критеріям

2.3.3. Корзина для покупок

2.3.3.1. Недоступно для користувачів, які не ввійшли в систему

2.4. Замовлення:

2.4.1. Оформлення замовлення

2.4.1.1. Додавання та вилучення продуктів у кошик продуктів

2.4.1.2. Вказати кількість, тип тощо. та підтвердження вибору

2.4.1.3. Перевірка системи на сумісність продукту і сповіщення в разі помилок

2.4.2. Скасування замовлення до зміни його статусу на поточний.

2.5. Сповіщення

2.5.1. Можливість отримувати сповіщення

3. Незареєстрований користувач:
   3.1. Реєстрація (надання персональних даних: логін, пароль, e-mail)
   3.2. Перегляд каталогів
   3.3. Пошук продуктів

2.1. Нефункціональні вимоги до системи

1.1. Інтерфейс – доступ до системи інтернет-магазину буде можливий через смартфон із системою Android.
1.2. Програма працюватиме на смартфонах з Android версії 5.0 / 5.1 Lollipop або новішої.
1.3. Програма дозволить здійснювати платежі в таких валютах: USD.
1.4. Програма повинна працювати на пристроях різного розміру. Від клітинок розміром - 1280 × 720 до клітинок розміром 1080 × 2280.
1.5. Програма зберігатиме не просто пароль, а й ключ, який буде використовуватися для доступу до системи. Шифрування буде здійснюватися за допомогою солі.

2.2.1. Опис ролі користувача

Діяльність, яку виконують певні групи користувачів, тобто діаграму варіантів використання. Головними дійовими особами є користувач і продавець.

2.2.2. Опис окремих випадків використання

1. Відредагуйте дані облікового запису.

Функція, за допомогою якої будь-який користувач може редагувати дані свого облікового запису, такі як ім'я користувача, пароль, аватар. Використовувалися команди Змінити ім’я користувача, Змінити пароль, Змінити аватар. Функція «Створити обліковий запис користувача»

викликається автоматично під час її створення, після чого користувач може викликати її будь-яку кількість разів (діапазон <0, + оо).

2. Змініть ім'я користувача.

Функція зміни логіну користувачем є частиною статусу Редагувати дані облікового запису. Змінює поточне ім’я користувача на нове, яке надав користувач у формі, створеній системою. Перед зміною нове ім’я підлягає перевірці (довжина та дозволені символи).

3. Змініть свій аватар.

Функція, яка є частиною статусу Редагувати дані облікового запису. З його допомогою користувач може змінити картинку свого профілю. Графічний файл, завантажений формою, після перевірки відповідності вимогам (формат, розмір) замінює поточний аватар користувача.

4. Змініть пароль.

Функція дозволяє користувачеві змінити пароль. Пароль, наданий користувачем, отриманий із форми, перевіряється системою перевірки щодо довжини, дозволених символів та додаткових умов безпеки, таких як необхідність містити принаймні одну велику літеру та один спеціальний символ (знак оклику, et / амперсанд, тощо).

5. Створіть обліковий запис користувача.

Функція, яка дозволяє користувачеві створити власний обліковий запис. Його використання призводить до відображення статусу Редагувати дані облікового запису. Після перевірки даних, введених користувачем у форму, система створює обліковий запис. Також запускається функція системи керування користувачами - надати користувачеві відповідні права.

6. Надайте відповідні права користувачеві.

Функція, за допомогою якої система керування користувачами надає створеному обліковому запису користувача відповідні права (ролі). За замовчуванням це дозволи людей, які використовують програму. Інші можливі ролі користувача: суперкористувач, адміністратор і власник.

Функція автоматично викликається, коли виконується функція Створити обліковий запис користувача.

7. Увійдіть.

Функція, за допомогою якої користувач входить в систему. Після натискання відповідної кнопки він буде перенаправлено на підсторінку, що містить форму входу. Після введення даних на основі вмісту відповідної таблиці в системній базі даних Система керування користувачами перевіряє правильність логіну та пароля (функція Verify User).

Після правильного введення даних користувач переходить до свого облікового запису, і таким чином отримує доступ до функцій, доступних для користувачів, які ввійшли в систему (різні для окремих ролей користувача). У разі неправильного введення логіна та пароля система виводить повідомлення про неправильні дані, надані користувачем.

8. Перевірте користувача.

Логічна функція, яка використовується для перевірки даних, наданих користувачем у формі входу, з даними, що містяться у відповідній таблиці системної бази даних (вона з’являється при виконанні функції «Вхід»). Якщо дані сумісні, вони повертають значення "true", інакше - значення "false".

9. Додати товари.

Функція, яка дозволяє продавцю додати новий товар в систему. Він може додавати нескінченну кількість продуктів. Після перевірки коректності товару, доданого у форму продавцем () - з'являються функції Редагувати опис товару, Редагувати категорію та Встановити суму оплати за товар.

10. Відредагуйте опис елемента.

Функція, за допомогою якої продавець редагує опис товару. Потім з’являється форма, в яку можна додати опис товару. Після правильної перевірки введеного опису він буде оновлено. Якщо надані дані не відповідають системним вимогам, відображається відповідне повідомлення.

11. Відредагуйте категорію товару.

Функція, яка дозволяє продавцю редагувати категорію даного товару. Система виводить форму, яку заповнює трейдер, потім – перевіряються дані.

Якщо перевірка успішно пройдена - категорія товару оновлюється, інакше система відобразить відповідне повідомлення.

12. Установіть суму оплати за товар.

Функція, завдяки якій продавець може встановити суму оплати за свій товар.

13. Відредагуйте вміст кошика.

Функція, яка використовується для редагування вмісту списку користувачем. Він складається з функції «Редагувати елементи в кошику». Він з’являється щоразу, коли ви додаєте новий елемент.

14. Редагувати товари в кошику.

Функція, яка є частиною підсистеми редагування вмісту кошика. Він полягає в додаванні або вилученні товарів, що містяться в кошику. Залежно від операції, яку виконує користувач, викликає функцію Додати товари в кошик, або - Вилучити товар з кошика та Змінити кількість придбаного товару.

15. Додайте товари в кошик.

Функція, за допомогою якої користувач може додати вибрані товари в кошик. Людина за допомогою пошукової системи знаходить товари, які її цікавлять. Коли продукт вибирається, він додається в кошик.

16. Витягніть товар з кошика.

Функція, яка дозволяє користувачеві видалити елемент зі списку, який редагується. Після його появи відображається список товарів у кошику. Потім покупець вибирає товар, який потрібно видалити, і видаляє його зі списку.

17. Змінити кількість придбаного товару.

Функція, яка дозволяє користувачеві редагувати кількість придбаних продуктів. Завдяки йому можна змінювати кількість товару. При натисканні на кнопку «плюс» кількість товарів збільшується, при натисканні на кнопку «мінус» – зменшується.

18. Додайте кредитну картку.

Функція, яка дозволяє покупцеві додати кредитну картку. Дана підсистема автоматично виконує функцію Редагувати дані кредитної картки. До облікового запису користувача можна додати лише одну картку.

19. Відредагуйте дані своєї кредитної картки.

Ця підсистема використовується для редагування даних кредитної картки, пов’язаних з обліковим записом користувача. У ньому представлена відповідна форма, яку має заповнити покупець. Після первинної перевірки формату наданих даних (номер картки, номер CVV / CVC) з'являється функція Перевірити дані кредитної картки. У разі неправильного введення інформації відображається відповідне повідомлення.

Якщо дані введені правильно, кредитна картка буде присвоєна обліковому запису користувача. В іншому випадку з’явиться інформативне повідомлення про несумісність його даних.

20. Перевірте дані своєї кредитної картки.

Логічна функція перевіряє відповідність реквізитів кредитної картки інформації, яку має банк. Залежно від правильності введення інструкцій на екрані телефону з’явиться відповідне повідомлення.

21. Оплата готівкою при отриманні

Завдяки цій активності користувач може оплатити товар при доставці у кур’єра або на пошті.

22. Перевірте наявність коштів на кредитній картці.

Використовуючи вищезгадану функцію, користувач може перевірити, чи зазначена сума на даній кредитній картці. Подібно до перевірки даних картки з'являється інформація про хід перевірки коштів.

23. Оцініть товар.

Функція, за допомогою якої покупець може оцінити даний товар. Після натискання користувачем відповідної кнопки біля вибраного товару до чисельника його рейтингу в базі даних додається значення. Рейтинг товару регулярно оновлюється. Клієнт може залишити лише один відгук про продукт. Його також можна змінити або скасувати в будь-який час.

2.3. Сценарії використання

У цьому розділі описані такі варіанти використання: «Створити обліковий запис користувача», «Увійти», «Змінити кількість придбаних товарів», «Перевірити дані кредитної картки», «Додати товари». «Створити обліковий запис користувача», «Увійти» та «Перевірити дані кредитної картки». Їх вибрали тому, що вони є важливими функціями для користувача, які дозволяють використовувати всі можливості інтернет-магазину.

Таблиця 2.1 – Випадок використання: «Створити обліковий запис користувача»

Назва	Створення облікового запису користувача.
Опис	Функція, за допомогою якої користувач створює свій обліковий запис.

Продовження таблиці 2.1

Загальне	Користувач, система керування користувачами.
Пріоритет	Високий - необхідний для доступу до сайту.
Передумови	-
Остаточні умови	У системі створюється обліковий запис користувача, що містить його дані.
Основний потік подій	1. Користувач відвідує сторінку реєстрації. 2. Система відобразить форму редагування даних облікового запису. 3. Користувач заповнює форму. 4. Система підтверджує правильність наведених даних (валідація). 5. Система створює обліковий запис користувача. 6. Система оновлює дані користувача. 7. Система керування користувачами надає користувачеві відповідні права (тип облікового запису).

Закінчення таблиці 2.1

Альтернативний перебіг подій	4а. Система виявляє, що надані дані неправильні. 5а. Система виводить повідомлення про неправильні дані. 6а. Поверніться до кроку 2.
Особливі вимоги	1. Час виконання функції не може перевищувати 3 секунд. 2. Висока надійність функції – коефіцієнт POFOD не може перевищувати значення 0,0001 (на кожні 10 000 виконання функції може бути не більше однієї помилки).

Таблиця 2.2 – Випадок використання: «Увійти»

Назва	Увійти.
Опис	Функція, за допомогою якої користувач входить у свій обліковий запис.
Загальне	Користувач, система керування користувачами
Пріоритет	Високий - необхідний для доступу до функцій користувача

Продовження таблиці 2.2

Передумови	Обліковий запис користувача створено.
Остаточні умови	Користувач входить у свій обліковий запис в системі.
Основний потік подій	1. Користувач переходить на сторінку входу. 2. Система відобразить форму входу. 3. Користувач заповнює форму входу. 4. Система керування користувачами успішно перевіряє існування користувача з заданим логіном та паролем у базі даних. 5. Система перенаправляє користувача на головну сторінку сайту.

Закінчення таблиці 2.2

Альтернативний перебіг подій	4а. Система керування користувачами виявила невідповідність пароля. 5а. Система відображає повідомлення про невідповідність пароля. 6а. Поверніться до кроку 2 4б. Система керування користувачами стверджує, що немає користувача з даним логіном. 5б. Система виводить повідомлення про те, що користувача з даним логіном немає. 6б. Поверніться до кроку 2
Особливі вимоги	1. Час відповіді сервера не може перевищувати 3 секунди. 2. Висока надійність функції – коефіцієнт POFOD не може перевищувати значення 0,0001 (на кожні 10 000 виконання функції може бути не більше однієї помилки).

Таблиця 2.3 – Випадок використання: «Змінити кількість придбаних товарів»

Назва	Змінити кількість придбаних товарів.
Опис	Функція, за допомогою якої користувач редагує кількість товарів у кошику.
Загальне	Користувач
Пріоритет	Низький
Передумови	Користувач увійшов в систему. Товар є в кошику.
Остаточні умови	Кількість, які продуктів редагується.
Основний потік подій	1. Користувач відвідує сторінку, що містить його продукти. 2. Користувач натискає кнопку «Плюс або мінус». 3. Система збільшує або зменшує кількість продуктів. 4. Система підтверджує правильність наведених даних (перевірка - щоб кількість була не менше 0 і більше 8).

Закінчення таблиці 2.3

Альтернативний перебіг подій	4а. Система виявляє, що надані дані неправильні. 5а. Система виводить повідомлення про невірно введені дані.
Особливі вимоги	1. Час виконання функції сервером не може перевищувати 2 секунд. 2. Висока надійність функції – коефіцієнт POFOD не може перевищувати значення 0,0001 (на кожні 10 000 виконання функції може бути не більше однієї помилки).

Таблиця 2.4 – Випадок використання: «Перевірка даних кредитної картки»

Назва	Перевірка даних кредитної картки.
Опис	Функція, за допомогою якої система перевіряє правильність даних кредитної картки, введених користувачем.
Загальне	Система управління платежами
Пріоритет	Високий - необхідний для обробки платежів

Закінчення таблиці 2.4

Передумови	Картка була додана користувачем, дані картки були заповнені користувачем.
Остаточні умови	Дані кредитної картки користувача перевіряються системою.
Основний потік подій	1. Система завантажує дані кредитної картки з облікового запису користувача. 2. Система перевіряє, чи є карта з параметрами, заданими користувачем. 3. Система підтверджує правильність даних картки.
Альтернативний перебіг подій	3а. Система виявляє, що дані картки неправильні. 4а. Система надсилає на обліковий запис користувача повідомлення про неправильні дані картки.
Особливі вимоги	1. Час виконання функції не може перевищувати 5 секунд. 2. Висока надійність функції – коефіцієнт POFOD не може перевищувати значення 0,000

Таблиця 2.5 – Випадок використання: "Додати товар"

Назва	Додати товар
Опис	Функція, за допомогою якої продавець додає новий товар.
Загальне	Продавець
Пріоритет	Низький
Передумови	-
Остаточні умови	Новий продукт додано на сайт.
Основний потік подій	1. Продавець відвідує сторінку для додавання товару. 2. Продавець вибирає файл для завантаження з диска. 3. Система відображає форму видання даних про продукт. 4. Продавець заповнює форму. 5. Система підтверджує правильність наданих даних.

Закінчення таблиці 2.5

Альтернативний перебіг подій	5а. Система виявляє, що надані дані неправильні. 6а. Система виводить повідомлення про невірно введені дані 7а. Поверніться до кроку 3.
Особливі вимоги	1. Час виконання функції сервером не може перевищувати 3 секунд. 2. Висока надійність функції – коефіцієнт POFOD не може перевищувати значення 0,0001 (на кожні 10 000 виконання функції може бути не більше однієї помилки).

2.4. Концептуальна схема класів

У цьому розділі наведено концептуальну схему класів та опис. Опис класів містить у собі, для чого призначені класи та які їхні методи найважливішими.

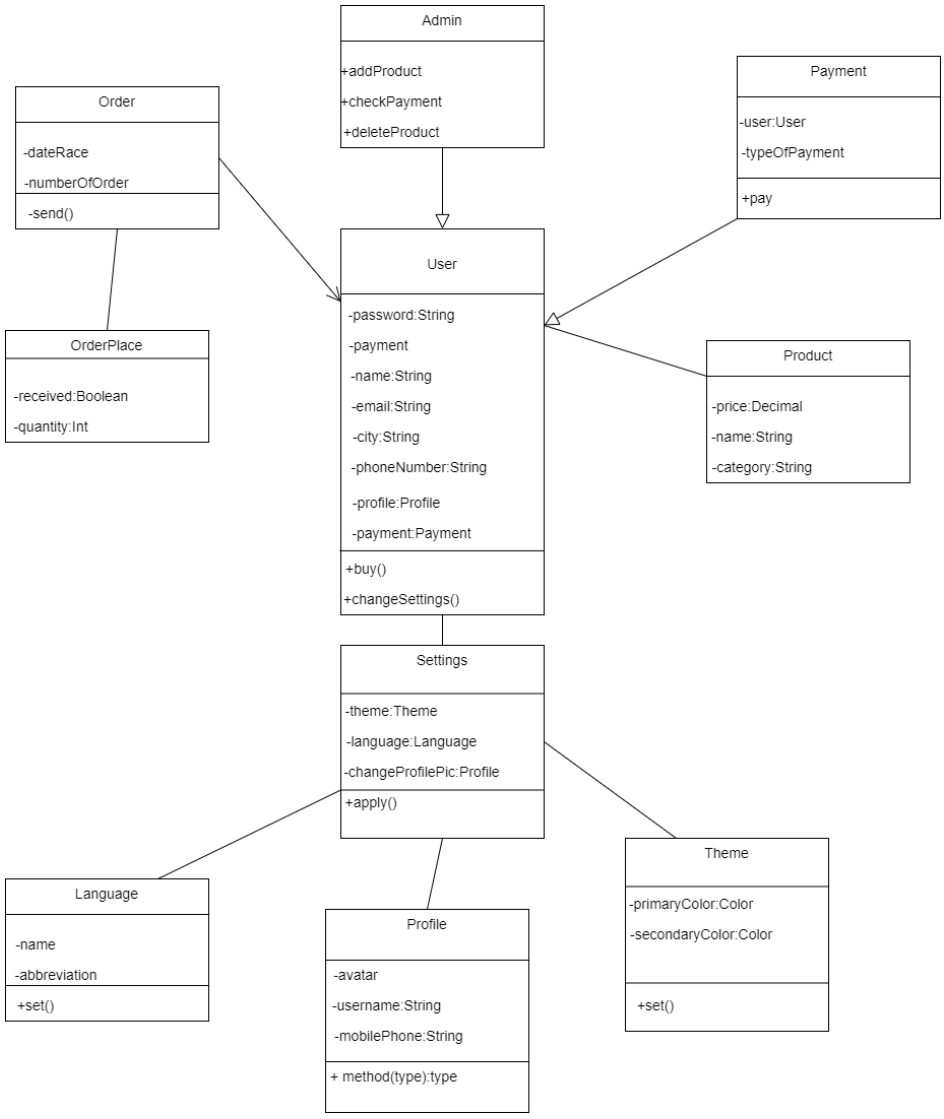

Рисунок 2.1 – Концептуальна діаграма класів

Опис занять:

1. Admin - клас використовується для опису можливостей адміністратора, які функції він виконує і за що відповідає. Перша функція - addProduct - функція відповідає за додавання товару в магазин, checkPayment - дозволяє переглянути товари в замовленні і перевіряє, чи оплачене замовлення, deleteProduct – видаляючи заданий товар з інтернет-магазину.

2. Користувач – клас використовується для опису користувача. Він містить поля, в яких зберігається інформація про персональні дані користувача. У класі є методи: купити () і змінить налаштування (). Метод buy () відповідає за логіку покупок. Метод ChangeSettings () - дозволяє користувачеві змінити мову програми, тему та дані профілю, такі як: аватар, ім'я або номер телефону.

3. Замовлення - клас призначений для зберігання даних про замовлення. У ньому є два поля, що відповідають за час отримання товару та його позицію. Клас також містить метод send () для відправки замовлення.

4. Оплата – клас використовується для опису оплати товару. Він має два поля, user - поле, в якому зберігаються дані про користувача, який оплатив замовлення і typeOfPayment - поле, в якому зберігаються вказівки щодо способу оплати. Він має спосіб оплати, який використовується для оплати користувачем.

5. OrderPlace - клас призначений для зберігання даних про позицію замовлення. Клас містить два поля: отримано - перевіряє, чи отримав клієнт товар, а кількість - відповідає за кількість замовлень.

6. Product - клас використовується для опису продуктів. Він містить три поля: ціна - ціна товару, назва - його назва, категорія - категорія товару для полегшення його пошуку.

7. Налаштування - клас призначений для опису дозволів. Він має такі поля, як theme, яка відповідає за тему програми, language, яка відповідає за мову програми, та changeProfilePic, яка відповідає за зберігання даних про

користувача. За застосування вибраних налаштувань відповідає метод apply ().

8. Мова - клас використовується для встановлення мови користувачем. Містить метод set (), який дозволяє користувачеві застосувати мову.

9. Профілі - зберігання даних, які будуть відображатися користувачеві. Клас містить три поля: avatar - зберігання зображення профілю користувача, ім'я користувача - що містить ім'я користувача, mobilePhone - номер телефону та метод (тип) метод, який використовується для перевірки типу дозволів: користувач або адміністратор.

10. Тема - Зберігайте тему. Клас містить два поля: primaryColor - це світла тема програми, secondaryColor - темна тема. За налаштування теми відповідає метод set ().

2.5. Діаграма ERD

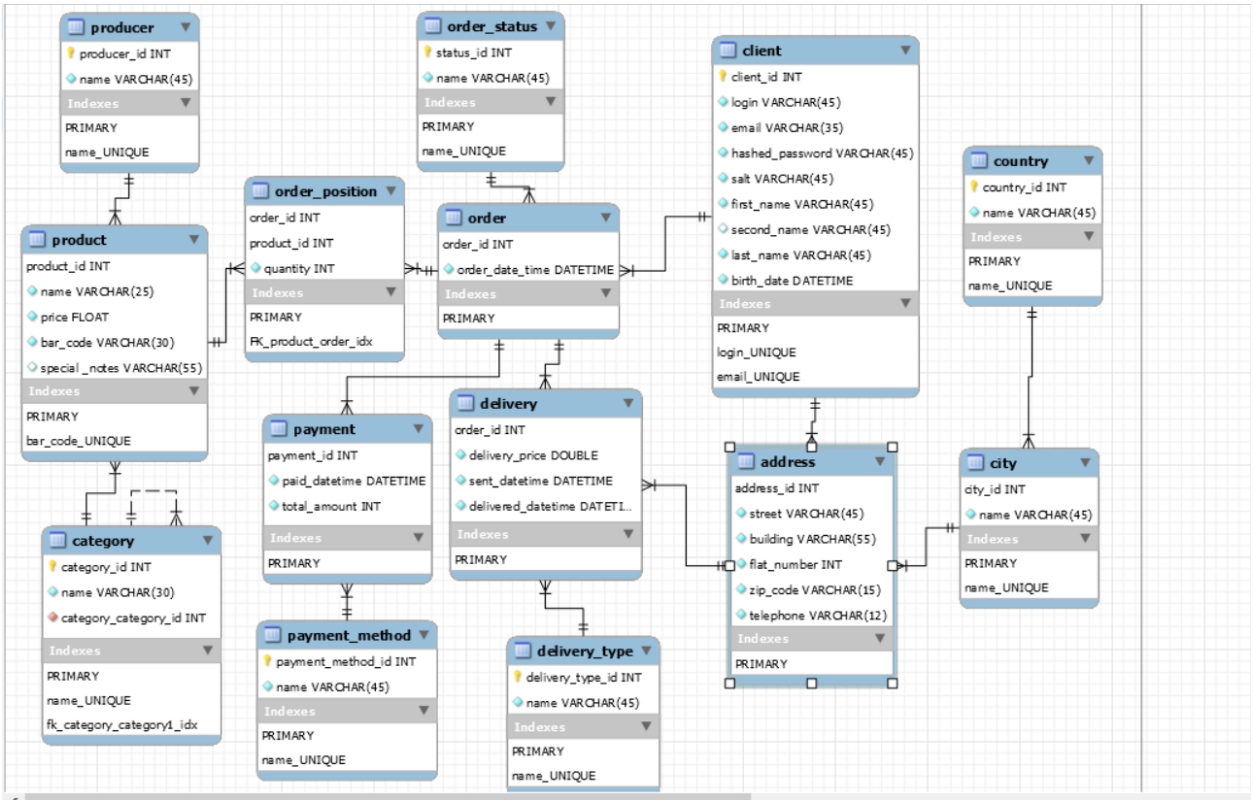

Рисунок 2.2 – Діаграма ERD бази даних

Опис окремих об'єктів:

Product:

У Застосунку можна придбати продукти, дані яких зберігаються в базі даних. Об’єкт продукту представляє товар для продажу та створює зв’язки з об’єктами категорії продукту Category та об’єктом даних виробника Producer.

Category:

Категорія дозволяє розділити продукти на підкатегорії, щоб полегшити їх пошук.

Producer:

Суб'єкт, що представляє дані виробників товарів.

Client:

Клієнти, чиї дані зберігаються в базі даних, мають базову інформацію, необхідну для перевірки (таку як: ім’я, прізвище, електронна пошта).

Додаткова інформація про клієнта, наприклад дата народження, дозволяє обмежити доступ до матеріалів, призначених для дорослих.

Address:

Дані про місцезнаходження потрібні, щоб перевірити, чи можна доставити товар до певного міста чи країни.

Order:

Суб'єкт, що зберігає замовлення, містить два поля: час замовлення та ідентифікатор клієнта, який замовив товар, а також статус (товар ще в магазині чи готовий до забору).

Order status:

Таблиця зі статусом замовлення. За допомогою поля status_id ми можемо перевірити, на якому етапі реалізації знаходиться замовлений товар.

Order position:

Таблиця, що містить кількість замовлень і позицій для кожного замовлення.

Country:

У таблиці показано країни, з яких реєструються користувачі.

City:

Таблиця міст, з яких здійснюється реєстрація.

Delivery:

Таблиця, яка дає змогу перевірити дату та час початку та закінчення доставки.

Delivery type:

Таблиця, в якій зберігається інформація про те, як буде доставлений товар. Він включає вид доставки.

Payment:

Таблиця з інформацією, необхідною для здійснення платежів. Він містить такі дані, як: сума до сплати, час платежу та тип платежу.

Payment method:

Таблиця, що містить обраний спосіб оплати товару.

2.6. Технології та інструменти, які використовуються для побудови програми

У цьому розділі описано інструменти розробки застосунків Android. Він містить інформацію про середовище розробки, мову програмування, базу даних і можливості цієї бази даних і репозиторію коду.

2.6.1. Виробниче середовище

Android Studio — це середовище на основі IntelliJ, в якому як початківці, так і досвідчені розробники можуть писати та розробляти свої програми. Android Studio також дозволяє налагоджувати, працювати з git в інтегрованому терміналі та з кодом: доповнювати його та підключати до відповідних бібліотек.

2.6.2. Мова програмування

Kotlin - статично типізована мова програмування, що працює на віртуальній машині Java, яка розробляється розробниками JetBrains. Kotlin реалізований на платформі Android. Її було оголошено офіційною мовою програмування на конференції Google I/O 2017 .

2.6.3. База даних

База даних виконана за технологією Firebase . Дана система розроблена компанією Firebase Inc. У 2011 році була створена платформа, яка дозволяє створювати мобільні та інтернет-додатки. База даних Firebase має тип бази даних NoSql.

Firebase пропонує такі можливості:

1. Авторизація - цей модуль відповідає за визначення методів авторизації користувачів. У нашому розпорядженні є вхід через електронну пошту, Facebook, Twitter та багато інших варіантів.

2. База даних - відповідає за зберігання даних, введених користувачами, наприклад, даних, необхідних для відправлення замовленого товару. Firebase надає два типи баз даних: база даних реального часу та Cloud Firestore.

3. Сховище – завданням цього модуля є зберігання файлів. Користувач може використовувати програму для завантаження різних файлів у проект Firebase, наприклад файлів із розширенням PDF.

4. Хостинг - дозволяє реалізувати як простий веб-сайт, так і складну інтернет-Застосунку.

5. Функціональність – дозволяє запускати певний код на стороні Firebase без необхідності використання зовнішнього сервера.

6. ML Kit - дозволяє використовувати нейронні мережі для вирішення завдань в програмах для Android та iOS.

У проекті використовується Spark Plan - це базовий план, який дозволяє створити 10 безкоштовних проектів. Ви можете перевірити, чи приверне створена програма одержувачів чи ні. Цей план дозволяє наступне:

- авторизація - цей модуль відповідає за визначення методів авторизації користувачів. У нашому розпорядженні є вхід через електронну пошту, Facebook, Twitter та багато інших варіантів;
- Cloud Firestore - це гнучка і масштабована база даних, яка зберігає дані в документах. Їх, у свою чергу, можна збирати в колекції;
- хмарні функції – у нас є можливість виконувати код функції, яку ми написали у хмарному середовищі йому не потрібна ні інфраструктура, ні серверне читання;
- хостинг - дозволяє реалізувати як простий веб-сайт, так і складну інтернет-Застосунку;
- Firebase ML – пакет SDK для мобільних пристроїв, який надає досвід машинного навчання Google у додатках Android та iOS у потужному, але простому у використанні пакеті;
- база даних реального часу – цю базу даних можна розглядати як об’єкт JSON, що зберігається в хмарі. У нас тут немає таблиць чи записів. Після додавання даних до бази даних вони стають частиною збереженої структури, яка містить ключ ідентифікації;
- хмарне сховище - дозволяє зберігати мультимедійні дані;
- Test Lab – цей сервіс дозволяє тестувати мобільні застосунки.

Firebase має наступні переваги:

- Google Cloud - віртуальний диск для зберігання та обміну даними. Авторизація - дозволяє реєструватися через Gmail, Facebook, Twitter;
- доступність у кількох системах;

- оновлення в режимі реального часу;
- безкоштовно;
- швидкість розвитку.

Слабкі сторони Firebase:

- немає провідника даних;
- немає вбудованої авторизації.

2.6.4. Репозиторій коду

Проект використовує репозиторій вихідного коду Git на сервері GitHub. GitHub - хостинг-сайт, присвячений програмуванню проектів за допомогою системи контролю версій Git.

Функціональність GitHub:

- Bug tracker - програма для реєстрації та керування інформацією про помилки, що зустрічаються в програмному забезпеченні;
- розвилка - ситуація, коли розвиток проекту вже не веде один маршрут, а розбивається на дві або більше гілок;
- Pull-запити – особа з форком може подати свій код для прив'язки до головного сховища;
- організації розробників, що працюють над репозиторіями.

Висновки до розділу 2

У даному розділі було розглянуто моделювання та проектування майбутнього застосунку. А також, ознайомлення з засобами, які будуть використанні для створення мобільного застосунку.

3. ПРОГРАМНА РЕАЛІЗАЦІЯ ЗАСТОСУНКУ

У цьому розділі описуються послідовні елементи програми, які були реалізовані щоб побудувати систему. Також представлено вихідний код запущеної програми.

3.1. Структура бази даних

В ході впровадження база даних була змінена з реляційної на нереляційну – Firebase. В ході реалізації архіви були перетворені з типу SQL - (це свого роду реляційні бази даних, тобто багато таблиць даних можуть взаємодіяти одна з одною, вони пов'язані одна з одною), до NoSQL - нереляційної бази даних. , дані зберігаються у вигляді пари об'єктів «ключ-значення», у яких між ними немає реляційних зв'язків.

Нереляційна база даних була обрана, оскільки вона обробляє дані швидше, ніж реляційна. Бази даних NoSQL часто швидші, оскільки їх моделі простіші, великі системи NoSQL можуть бути достатньо гнучкими, щоб краще дозволити розробникам використовувати програми у спосіб, який відповідає їхнім потребам.

База даних складається з 5 колекцій: «CartList» показано на рисунку 3.2 і містить дані кошика користувача, "Maded" можна побачити на рисунку 3.4. У ньому є дані про оброблені замовлення, а це означає, що вони готові до отримання, «Users» включає дані про користувачів програми – рисунок 3.5.

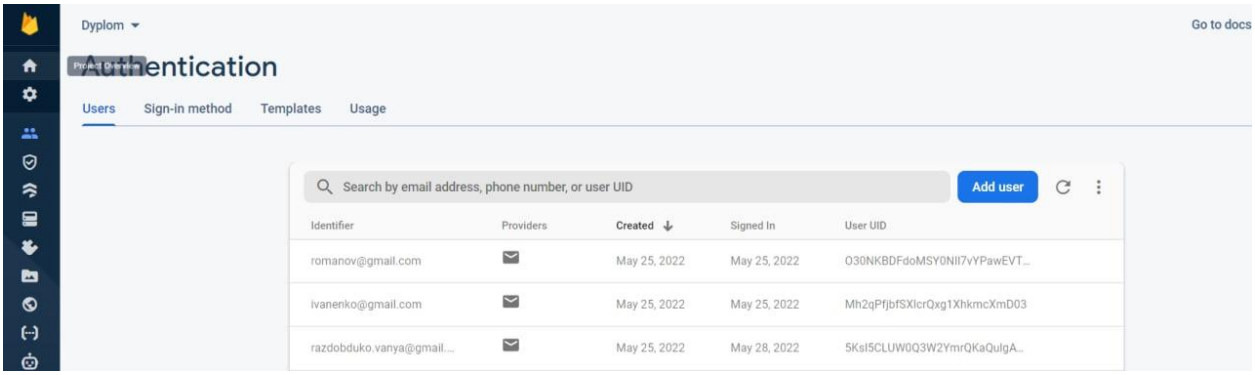

Рисунок 3.1 – Список зареєстрованих користувачів

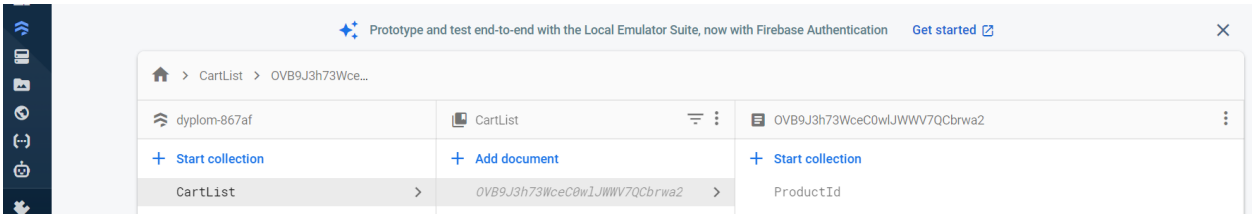

Рисунок 3.2 – Список кошика для збору

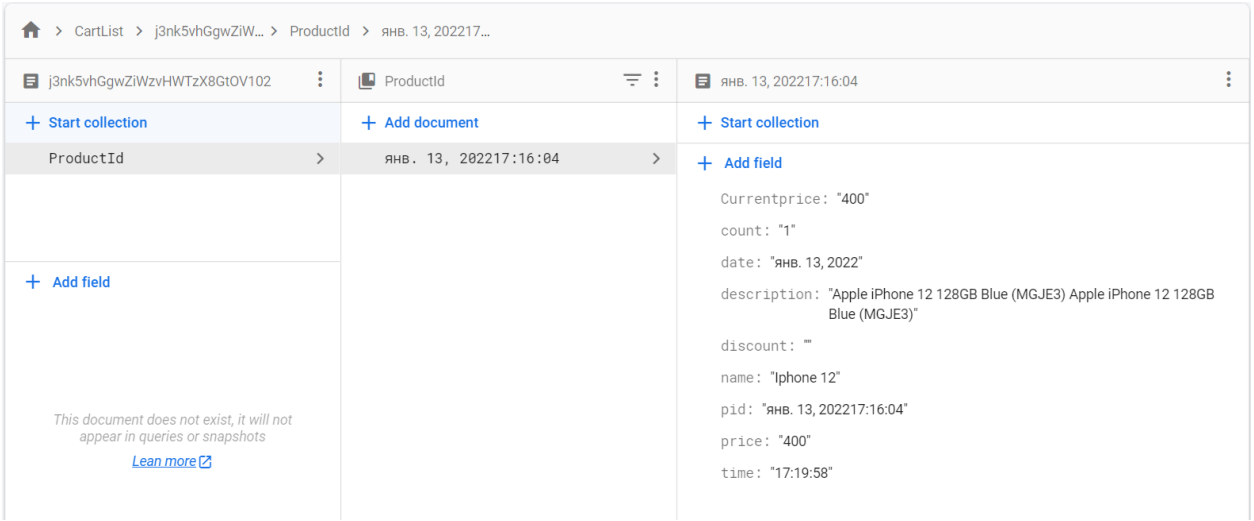

Рисунок 3.3 – Документи колекції ProductId

Колекція "CartList" містить документ під ім'ям ідентифікатора користувача. Він також включає колекцію ProductId, яка складається з елементів, доданих користувачем у кошик.

У колекції ProductId є документи, які містять інформацію про товар у кошику. Кількість товару відображається в полі «Кількість», спочатку встановлено 1. Повідомлення про зміну кількості виводиться в прямому ефірі.

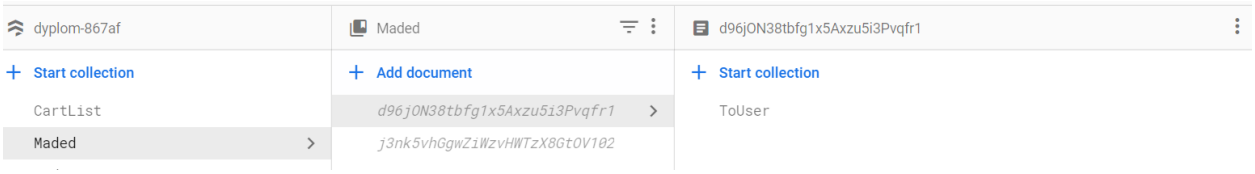

Рисунок 3.4 – Виготовлена колекція

Для доставки замовлень використовується колекція «Maded». Він містить документи, які мають ідентифікатор користувача як ідентифікатор, тому створюється колекція ToUser.

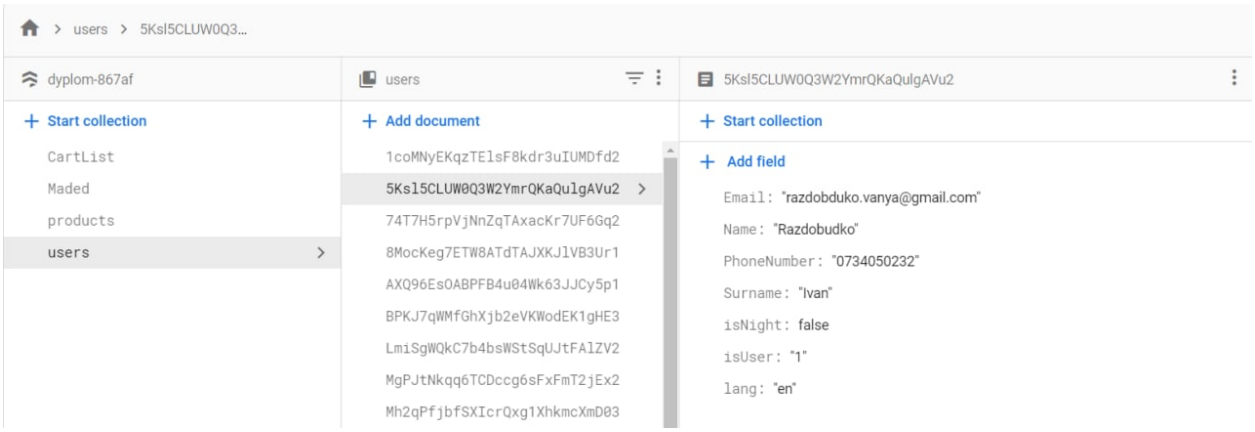

Рисунок 3.5 – Колекція користувачів

Колекція «ToUser» містить вихідні матеріали з бази замовлень. Після видалення елементів із колекції «Замовлення» вони будуть збережені в базі даних «Зроблено».

Колекція «Замовлення» містить перелік документів, які є замовленнями користувача. Документ отримує ідентифікатор користувача як ім’я та зберігає інформацію про замовлення: адреса, введена користувачем, ціна в залежності від кількості товарів. Їх значення складають і дані користувача - ім'я, номер телефону, час, тобто коли було зроблено замовлення.

Колекція «Products» бере дату та час додавання продукту як ідентифікаційних документів. Кожен продукт описується та містить посилання на зображення, яке розміщується в Firebase Storage.

Колекція «Users» бере назву документів та ідентифікатор створених користувачів і містить інформацію, взяту з реєстраційної діяльності. Поле «isUser» не надається користувачем. Він відповідає за те, чи є користувач адміністратором. Якщо в полі «isUser» встановлено значення 1, це означає, що це звичайний споживач, якщо «isUser» = 0, то цей користувач є адміністратором.

Список зображень у Firebase Storage містить окремі значки. Кожен з них має назву, яка є датою додавання продукту до бази даних.

3.2. Модуль реєстрації

На наступному рисунку показано перегляд фрагмента авторизації:

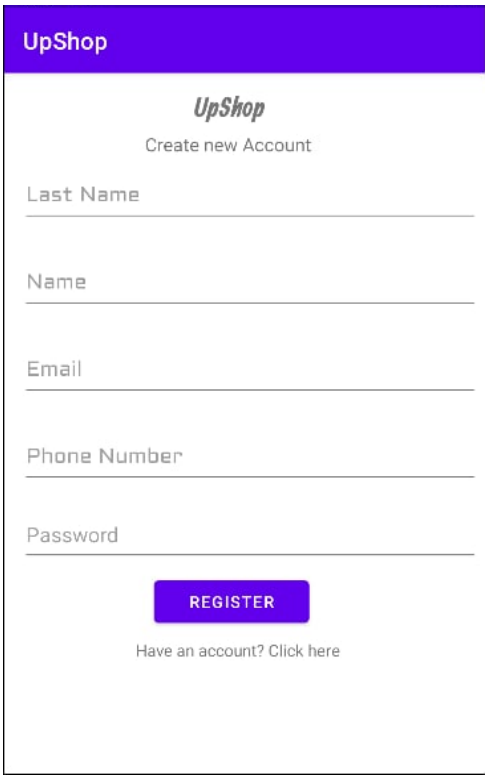

Рисунок 3.6 – Вигляд фрагменту авторизації

На рисунку 3.6 користувач бачить назву програми та поля, які необхідно заповнити, ввівши такі дані, як: ім’я та прізвище, адреса електронної пошти та пароль. Після натискання кнопки «Зареєструватися», якщо персональні дані введено правильно, відповідна особа переходить на головну сторінку. Користувач може перейти на сторінку входу, натиснувши на текст "Маєте обліковий запис? Натисніть". Цей витяг відповідає функціональній вимозі - 2.1.1.2 Реєстрація. Фрагменти сценарію, пов’язані з реєстрацією до програми, знаходяться у файлі «Автентифікація» в класі під назвою «Реєстрація»

```
fAuth.createUserWithEmailAndPassword(email,password).addOnCompleteListener(this) {task->
        onComplete(task,Name,Surname,phoneNumber,email)
        buildCategories(Name,Surname,phoneNumber,email)
    }
```

Рисунок 3.7 – Перелік коду: створення нового користувача

Коли відповідна особа реєструється за допомогою форми заявки, програма перевіряє, чи правильний електронний лист (він повинен мати спеціальний символ @, а не починатися з: символів "." тощо). Пароль (повинен бути більше 5 символів). Ми створюємо нового користувача за допомогою методу createUserWithEmailAndPassword, він містить ще два методи: onComplete (), який показано на рисунку 3.7 і buildCategories () – рисунок 3.8.

```
private fun onComplete(
    task: Task<AuthResult>,
    Name: String,
    Surname: String,
    phoneNumber: String,
    email: String
) {
    if(task.isSuccessful){

        //send verification link
        verificationEmail()
        progressBar.visibility = View.VISIBLE
        Toast.makeText( context: this, text: "User Created",Toast.LENGTH_SHORT).show()
        userId = fAuth.currentUser!!.uid
        var documentReference:DocumentReference = fStore.collection( collectionPath: "users").document(userId)

        var user:HashMap<String,Any?> = buildCategories(Name,Surname,phoneNumber,email)

        documentReference.set(user).addOnSuccessListener {  it: Void!
            Log.d( tag: "TAG",  msg: "onSuccess: user profile is created for$userId")
        }
        startActivity(Intent(applicationContext,MainActivity::class.java))

    }else{
        Toast.makeText( context: this, text: " ERROR! " + task.exception!!.message,Toast.LENGTH_SHORT).show()
    }
}
```

Рисунок 3.8 – Перелік коду : функція OnComplete

Функція відповідає за введення даних з форми в базу даних, зберігаються такі значення: id - унікальний для кожного користувача, ім'я, прізвище, номер телефону, email. У разі правильно введених даних

користувач перенаправляється на головну сторінку програми, у разі неправильно введених даних користувач отримує повідомлення про те, в чому він допустив помилку.

```
private fun buildCategories(Name: String,
                            Surname: String,
                            phoneNumber: String,
                            email: String):HashMap<String,Any?>{
    return hashMapOf(
        "Name" to Name,
        "Surname" to Surname,
        "Email" to email,
        "PhoneNumber" to phoneNumber,
         "isUser" to "1"
    )
}
```

Рисунок 3.9 – Перелік коду: функція buildCategories

Сценарій функції buildCategories бере дані з форми та вставляє їх у поля бази даних.

3.3. Модуль входу

Логін створено на основі введення правильної адреси електронної пошти та пароля. Якщо вони правильні, користувач отримує повідомлення про успішний вхід. Однак якщо особисті дані були неправильними, відповідна особа отримує повідомлення про те, що вони неправильні. Перевірка даних була підготовлена за допомогою Firebase і методу, представленого на Рисунок 3.13.

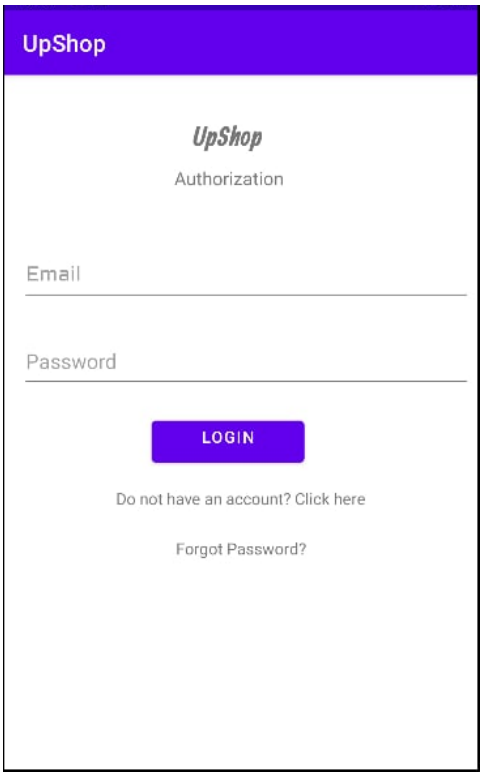

Рисунок 3.10 – Сторінка входу

На сторінці входу є два поля, які необхідно заповнити, ввівши дійсну адресу електронної пошти та пароль. Натиснувши на кнопку «Увійти», користувач переходить на головну сторінку програми. «Забули пароль» використовується для введення нового пароля. Ця команда викликає функцію sendPasswordResetEmail (). Його діяльність показана на рисунку 3.11. Функціональна вимога дотримана: 2.1.1.1. Увійти.

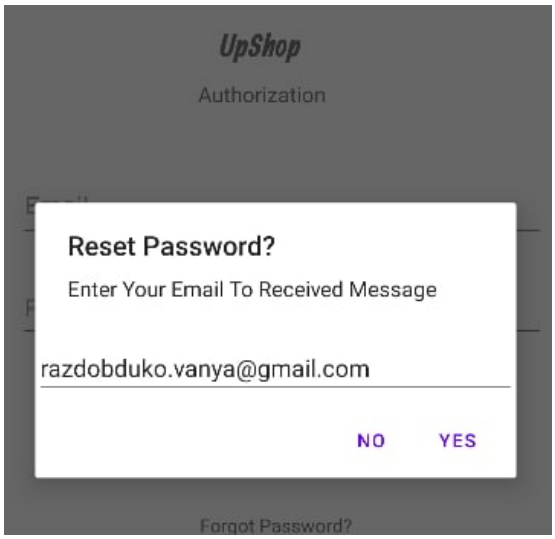

Рисунок 3.11 – Введення нового пароля

Користувач отримує повідомлення про те, що він повинен ввести адресу електронної пошти, щоб отримати повідомлення про новий пароль, у відповідної особи є два варіанти: «Так» або «Ні».

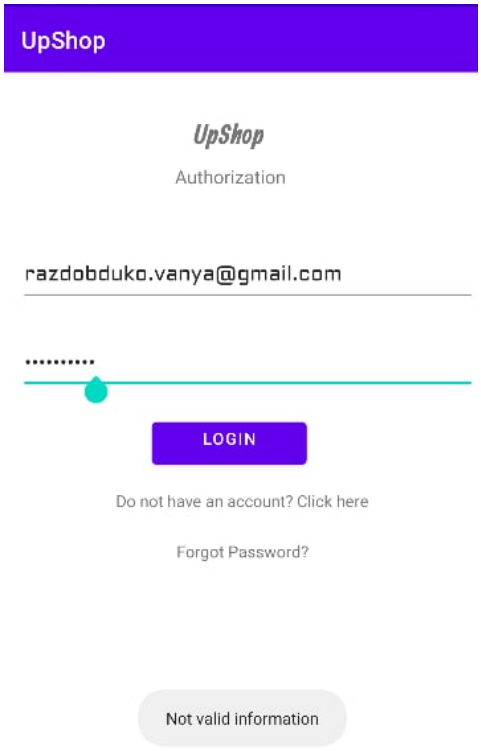

Рисунок 3.12 – Невірні дані

```
    fAuth.signInWithEmailAndPassword(email, password).addOnSuccessListener{
        task ->
         task.user?.let {  it: FirebaseUser
             progressBar.visibility = View.VISIBLE
             checkUserLevel(it.uid) }
    }
         .addOnFailureListener{  it: Exception
             Toast.makeText( context: this, text: "Not valid information",Toast.LENGTH_SHORT).show()
         }
}
```

Рисунок 3.13 – Перелік коду: вхід у систему за допомогою Firebase

Перевіряє метод signInWithEmailAndPassword (електронна пошта, пароль) в базі даних, чи вірні дані. Якщо вони правильні, запускається метод addOnSuccessListener, а в разі помилки відображається метод addOnFailureListener.

3.4. Модуль домашньої сторінки

На рисунку 3.14 показано домашню сторінку з переліком продуктів. Він включає всі продукти, додані до програми. Спочатку виставляються старі товари, а потім нові. При натисканні на один із товарів користувач переходить на іншу сторінку, яка містить опис товару і дозволяє додати товар у кошик або купити його. Елементи на домашній сторінці додаються та відображаються асинхронно за допомогою команди GlobalScope, що означає, що користувач не повинен чекати відповіді від бази даних і може безпечно використовувати програму.

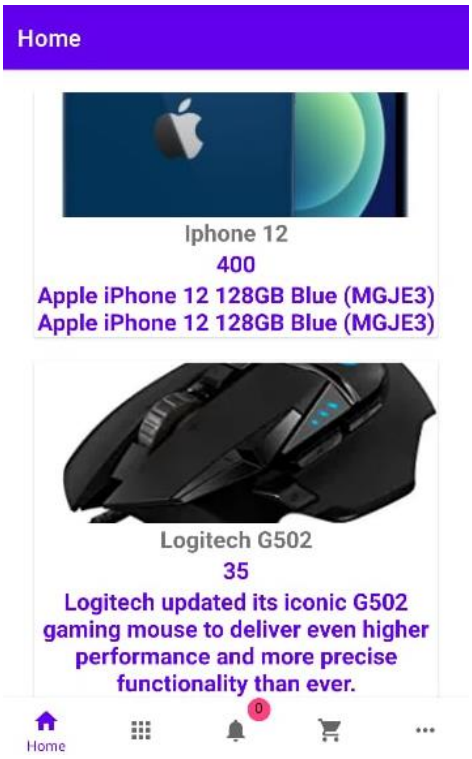

Рисунок 3.14 – Головна сторінка

Список продуктів на домашній сторінці відображається в компоненті RecyclerView - це один із компонентів системи Android. Він відповідає за відображення списку елементів із визначеним макетом (Layout). CardView відповідає за появу кожного елемента в списку. Створення інвентаризації рекомендованої продукції відповідає функціональним вимогам 2.1.3 і здійснюється автоматично, що відповідає п. 2.1.3.1.

```
val docRef = mDatabaseRef1.collection( collectionPath: "products")


docRef.get().addOnSuccessListener { documentSnapshot ->
    val city = documentSnapshot.toObjects<Products>()
    for (eachIndex in city.indices) {
        if (eachIndex != null) {
            mUploads.add(city[eachIndex])
        }
    }
    mAdapter = MyAdapter(requireActivity().applicationContext, mUploads)
    mRecyclerView.adapter = mAdapter
    //Log.d("VIBECHEdada", city[0].toString())
}
```

Рисунок 3.15 – Перелік коду : додавання елементів до адаптера

Наведений фрагмент коду знаходиться в кореневому каталозі HomeFragment.kt. Використовуючи колекцію Firebase, ми підключаємось до бази даних за допомогою методу collection (назва колекції), а за допомогою змінної, яка перетворюється на об’єкт класу Products, додаємо окремі список елементів до масиву, який відповідає за відображення цих елементів.

```
inner class MyViewHolder(itemView: View) : RecyclerView.ViewHolder(itemView){
    val productName = itemView.findViewById(R.id.product_name) as TextView
    val productPrice = itemView.findViewById(R.id.product_price) as TextView
    val description = itemView.findViewById(R.id.product_description) as TextView
    val imageView = itemView.findViewById(R.id.image_product) as ImageView
    lateinit var db:StorageReference

}
```

Рисунок 3.16 – Перелік коду : клас MyViewHolder

Клас MyViewHolder є внутрішнім класом типу MyAdapter. Вони знаходяться в каталозі адаптера. Клас MyAdapter успадковується від RecyclerView Adapter і приймає параметри: контекст типу Контекст і тип списку List - ця змінна відповідає за зберігання елементів з бази даних. Цей клас використовується для зберігання полів перегляду форми, в які будуть розміщені дані елементів зі списку.

```
override fun onBindViewHolder(holder: MyViewHolder, position: Int) {
    val products:Products = list[position]
    holder.productName.text = products.name
    holder.productPrice.text = products.price
    holder.description.text = products.description

    holder.db = FirebaseStorage.getInstance().getReference( location: "Product Images/")
    holder.db.child( pathString: "${products.pid}.jpg").downloadUrl.addOnCompleteListener{task->
        products.image = task.result.toString()
        Glide.with(context)
            .load(products.image)
            .into(holder.imageView.image_product)
    }
        .addOnFailureListener{ it: Exception
            Log.d( tag: "FailedCompleteListener",it.message.toString())
        }


    holder.imageView.setOnClickListener { it: View!
        val intent = Intent(holder.itemView.context,ProductDetailsActivity::class.java)
        intent.putExtra( name: "pid",products.pid)
        Log.d( tag: "Uniete",context.packageResourcePath)
        intent.flags = Intent.FLAG_ACTIVITY_NEW_TASK;
        holder.itemView.context.startActivity(intent)

    }
}
```

Рисунок 3.17 – Перелік коду: призначення значень

Функція відповідає за перезапис і відображення даних. Дані призначено з таблиці mUloads в об’єкт класу Product. Коли ви призначаєте значення полю products.image, відображаються зображення, збережені у FireStorage за допомогою Glide. Glide - ефективна бібліотека для управління мультимедіа та декодування і кешування в пам'яті та на диску для повторного використання. Його основна увага приділяється гладкому та ефективному відображенню, а також завантаженню та зміні зображень.

3.5. Модуль сповіщень

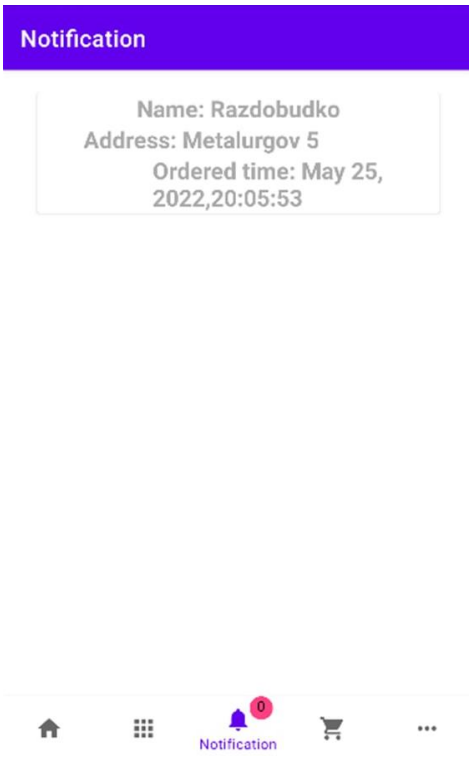

Рисунок 3.18 – Активність повідомлень

Користувач має можливість побачити кількість виконаних замовлень. Після переходу на фрагмент сповіщень користувач може переглянути попередні замовлення. Fragment Notifications відповідає за отримання інформації про статус замовлення. Він також служить історією бронювання. Це працює наступним чином: адміністратор після видалення замовлення зі списку (це означає, що замовлення вже знаходиться на місці доставки)

змінює поле в базі даних вибраного користувача, що відповідає статусу бронювання. Таким чином виконується вимога 2.5.1- можливість отримання повідомлень.

```
private fun checkSize(){
    var counter:Int = 0
    fr.collection( collectionPath: "Maded").document(uid).collection( collectionPath: "ToUser").whereEqualTo( field: "isChecked", value: 1)
        .get().addOnSuccessListener {  it: QuerySnapshot!
            counter = it.size()
            getItemCount(counter)
        }
}
```

Рисунок 3.19 – Перелік коду : завантаження кількості сповіщень

Функція checkSize () розташована в корені файлу MainActivity.kt і відповідає за отримання кількості елементів, які вже є на сторінці сповіщень, але ще не були переглянуті користувачем.

```
private fun updateBadgeCount(count: Int = 0){
    // Toast.makeText(this,"${count}",Toast.LENGTH_SHORT).show()

    val itemView = bottomNavigationView.getChildAt( index: 2) as? BottomNavigationItemView

    notificationBadges = LayoutInflater.from( context: this)
        .inflate(R.layout.badge_text,itemView, attachToRoot: true)
    notificationBadges?.notification_badge?.text = count.toString()

    bottomNavigationView.addView(notificationBadges)
}
```

Рисунок 3.20 – Перелік коду : оновлення кількості сповіщень

Метод updateBadgeCount () знаходиться в класі MainActivity в головному каталозі програми. Метод відповідає за оновлення кількості повідомлень, якщо в базі є новий елемент, то лічильник предметів збільшується до одиниці, якщо користувач перевірив усі сповіщення - номер лічильника змінюється на 0.

```
private fun checkFragmentVisibility() {
    val test: FavoritesFragment? = activity?.supportFragmentManager
        ?.findFragmentByTag( tag: "testID") as? FavoritesFragment
    if (test != null && test.isVisible) {
        Toast.makeText(requireContext().applicationContext,  text: "Check INgo", Toast.LENGTH_SHORT)
            .show()
    } else {
        //Toast.makeText(requireContext().applicationContext, "Didnds", Toast.LENGTH_SHORT).show()
        val docRef = fr.collection( collectionPath: "Maded").document(uid).collection( collectionPath: "ToUser")
        docRef.get().addOnSuccessListener { documentSnapshot ->
            val city = documentSnapshot.toObjects<Ordered>()
            for (eachIndex in city.indices) {
                if (eachIndex != null) {
                    val products: Ordered = mUploads[eachIndex]
                    fr.collection( collectionPath: "Maded").document(uid).collection( collectionPath: "ToUser")
                        .document(products.time.toString())
                        .update( field: "isChecked",  value: 0)
                }
            }
        }
    }
}
```

Рисунок 3.21 – Перелік коду: функція перевірки статусу фрагменту

Функція відповідає за перевірку статусу фрагмента: видимий чи ні, якщо користувач знаходиться на сторінці фрагмента «Повідомлення», ми завантажуємо дані з Firebase за допомогою змінної docRef: CollectionReference і встановлюємо значення поля, за яке відповідає чи перевірено елемент на 0, що означає, що користувач уже переглядав ці сповіщення.

3.6. Модуль кошика

Cart (Кошик) дозволяє видалити один товар, відредагувати кількість продуктів, видалити всі інгредієнти з кошика і перейти на сторінку оплати.

Налаштування кількості товарів здійснюється за допомогою elegantNumberButton.

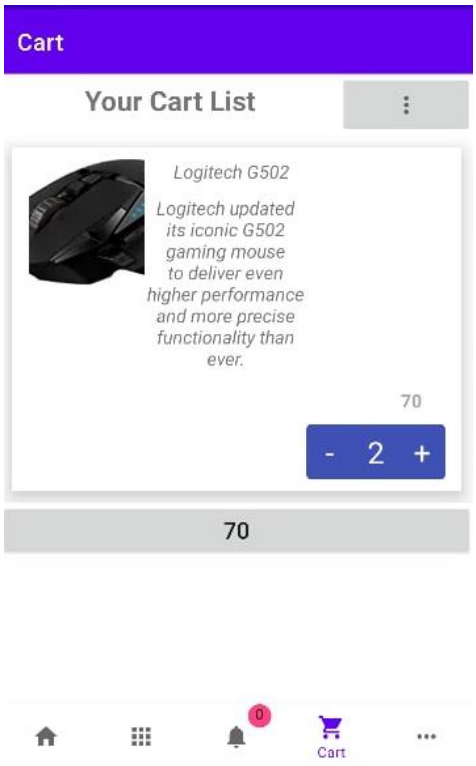

Рисунок 3.22 – Вид фрагменту Cart

Користувач при вході на сторінку кошика має можливість перевірити всю суму замовлення. На сторінці є кнопка, яка містить суму всіх позицій, завдяки якій користувач переходить до перегляду покупок. Кнопка підраховує всі продукти та їх кількість. Ви можете вибрати, скільки продуктів вам потрібно купити. На рисунку 3.22 показано дію та показано її завершення. У правому верхньому куті знаходиться кнопка, яка дозволяє вилучити всі товари з кошика (рисунок 3.25). Вимога 2.4.1.1 щодо додавання товарів у кошик та видалення виконано.

```
holder.description.text = productsCart.description


holder.elegantbutton.setOnValueChangeListener { view, oldVal, newVal ->
        // holder.productPrice =
         val num: String = holder.elegantbutton.number
         Log.d( tag: "Bra",holder.elegantbutton.number)
         val str = productsCart.price!!.toInt()
    Log.d( tag: "PriceS",str.toString())
         val result = num.toInt() * str!!.toInt()
         Log.d( tag: "Result",result.toString())
         holder.productPrice.text = result.toString()
         holder.db.collection( collectionPath: "CartList").document(holder.userId)
             .collection( collectionPath: "ProductId").document(productsCart.pid!!)
             .update( field: "Currentprice", result.toString())

         productsCart.Currentprice = result.toString()
    Log.d( tag: "CurrentPrice",productsCart.Currentprice.toString())

         productsCart.count = num.toString()
        // productsCart.price = result.toString()
         holder.db.collection( collectionPath: "CartList").document(holder.userId)
             .collection( collectionPath: "ProductId").document(productsCart.pid!!)
             .update( field: "count", num.toString())
}
```

Рисунок 3.23 – Перелік коду : підтримка кнопки визначення кількості замовлених продуктів

Фрагмент коду, який відповідає за обробку кнопки визначення кількості замовлених продуктів, можна знайти в каталозі адаптера, файлі CartAdapter.kt. Holder.elegantbutton.setOnValueChangeListener - відповідає за перевірку натискання кнопок «+» або «-». Якщо натиснути одну з цих кнопок, кількість (діапазон чисел від 0 до 8 включно) стягується з кнопки elegan, а також ціна з productCart.price, щоб мати можливість відобразити відповідну суму для кількість продуктів, обраних користувачем. Після розрахунку ми присвоюємо дані базі даних за допомогою функції оновлення. Ці дані необхідні для введення правильної ціни на всі товари.

Видалення окремих продуктів проводилося наступним чином:

```
    val builder = AlertDialog.Builder(activity)
    .setTitle("Card options")
    .setItems(option) { _, listener ->
        when(listener){
        0->GlobalScope.launch {  this: CoroutineScope
            holder.db.collection( collectionPath: "CartList").document(holder.userId)
                .collection( collectionPath: "ProductId").document(productsCart.pid!!)
                .delete()
                .addOnSuccessListener {  it: Void!
                    Toast.makeText(context,  text: "REMOVED", Toast.LENGTH_SHORT).show()
                }
                .addOnFailureListener {  it: Exception
                    Toast.makeText(context,  text: "Failed", Toast.LENGTH_SHORT).show()
                }
        }
            1->Toast.makeText(context,  text: "Adada",Toast.LENGTH_LONG).show()
    }
    }
    builder.create()
    builder.show()
}
```

Рисунок 3.24 – Перелік коду : видалення окремих товарів з кошика

Створено клас AlerDialog. Діалог — це елемент інтерфейсу, за допомогою якого можна значно розширити взаємодію з користувачем програми. Це дає нам можливість відображати короткі повідомлення, запитання користувачеві, запити прийняти рішення та багато інших альтернатив. Є два варіанти: «Видалити» та «Змінити» (опція, яка буде використовуватися в наступних версіях). При натисканні на кнопку «Видалити» елемент видаляється з бази даних за допомогою функції delete (), яка дозволяє видалити окремий документ із колекції. У цьому випадку документ можна знайти під записом productsCart.pid.

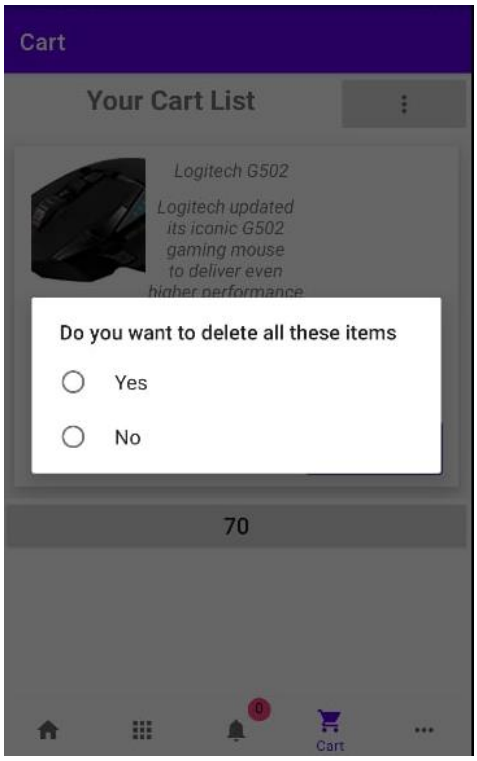

Рисунок 3.25 – Виймання всіх речей з кошика

На рисунку 3.25 користувач бачить повідомлення про те, що при натисканні кнопки всі елементи будуть видалені. Якщо натиснути кнопку, повідомлення не зникне і всі товари будуть в кошику. Видалення товарів з кошика схоже на рисунок 3.24.

3.7. Фрагментний модуль «Більше»

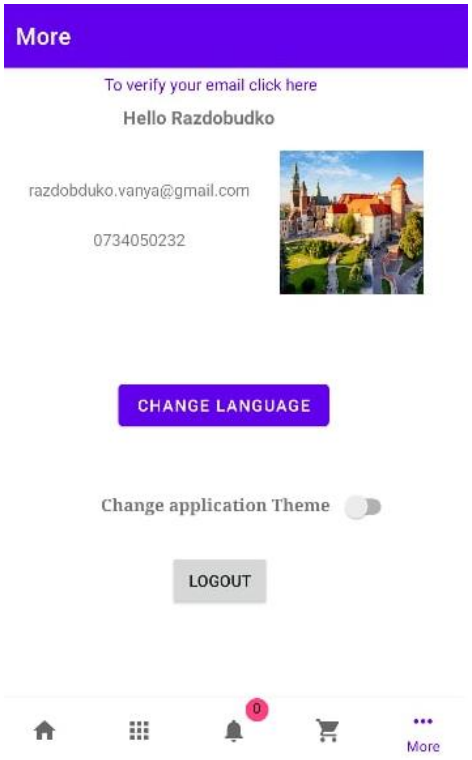

Рисунок 3.26 – Вид фрагменту «Більше»

На сайті користувач може побачити свої дані: електронну пошту, прізвище, номер телефону та встановлене зображення. Користувач може змінити мову програми завдяки кнопці «Змінити мову». На вибір є дві мови: англійська та українська. Це продемонстровано на рисунку 3.28. Представлений Switcher, за допомогою якого можна змінити тему. Коли вона вимкнена, можна побачити світлу тему, а коли її увімкнути – темну, як видно на рисунку 3.27. Вимоги 2.1.2.5 зміна мови, 2.1.2.6 зміна теми на темну виконано. Користувач має можливість вийти, натиснувши кнопку «Вийти».

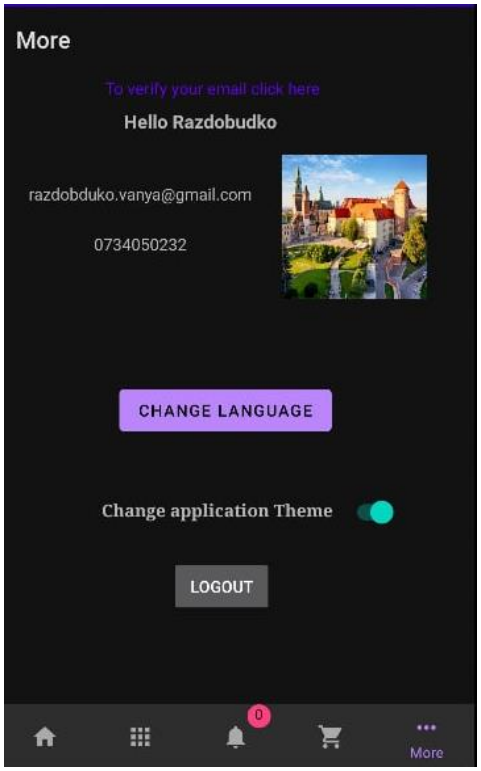

Рисунок 3.27 – Темна тема

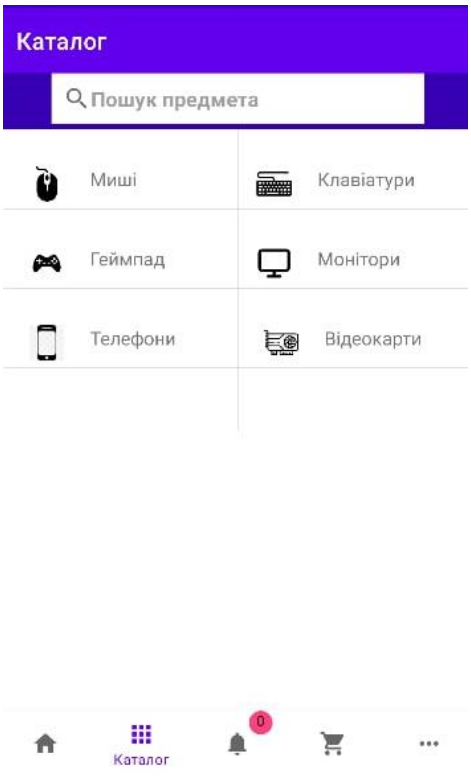

Рисунок 3.28 – Зміна мови

```
private fun changeColor() {
    val test: MoreFragment? = activity?.supportFragmentManager
        ?.findFragmentByTag( tag: "IDID") as? MoreFragment
    if (test != null && test.isVisible) {}
    else{
        mySwitch.setOnCheckedChangeListener { buttonView, isChecked ->
            if (isChecked) {

                // restartApp()

                AppCompatDelegate.setDefaultNightMode(AppCompatDelegate.MODE_NIGHT_YES)
                //sharedPref.setNightModeState(true)
                fStore.collection( collectionPath: "users").document(userId).update( field: "isNight",  value: true)

            } else if (!isChecked) {
                //restartApp()
                AppCompatDelegate.setDefaultNightMode(AppCompatDelegate.MODE_NIGHT_NO)
                fStore.collection( collectionPath: "users").document(userId).update( field: "isNight",  value: false)
                //sharedPref.setNightModeState(false)
            }
        }
    }
}
```

Рисунок 3.29 – Перелік коду : функція ChangeColor ().

Функція changeColor () розташована в корені файлу MoreFragment.kt. Він заснований на Switcher. Якщо ввімкнений, чорна тема встановлюється за допомогою команди AppCompatDelegate.setDefaultNightMode (AppCompatDelegate.MODE_NIGHT_YES). Зміни слід зберегти в базі даних, у полі користувача «isNight». Це працює таким чином, що кожна людина запам’ятовує власні налаштування.

```
    val docu: DocumentReference = fStore.collection( collectionPath: "users").document(userId)
    docu.addSnapshotListener { snapshot, e ->
        if (e != null) {
            Log.d( tag: "TAG",  msg: "Failed to read data from Firestore ${e.message}")
            return@addSnapshotListener
        }
        if (snapshot?.getBoolean( field: "isNight") == false) {
            mySwitch.isChecked = false
        }
        if (snapshot?.getBoolean( field: "isNight") == true) {
            mySwitch.isChecked = true
        }
    }

changeColor()
```

Рисунок 3.30 – Перелік коду : перевірка положення перемикача

Фрагмент сценарію, показаний на рисунку 3.30, знаходиться в кореневому каталозі, у файлі MoreFragment.kt. Після запуску програми слід запам'ятати позицію перемикача, але використовувати базу даних і перевірити, чи є поле «isNight» істинним чи хибним.

На вибір є дві мови: англійська та українська. Переклад виконується за допомогою Google Translate, у нас є strings.xml - для англійської та strings.xml (uk-rUK).

```
private fun languageCheck(){
    val documentReference: DocumentReference = fr.collection( collectionPath: "users").document(uid)
    documentReference.addSnapshotListener{snapshot, e->
        if (e != null) {
            Log.d( tag: "TAG",  msg: "Failed to read data from Firestore ${e.message}")
            return@addSnapshotListener
        }

        if(snapshot?.getString( field: "lang") == "en"){

            setLocale(snapshot.getString( field: "lang").toString())
        }
        else if(snapshot?.getString( field: "lang") == "uk"){

            setLocale(snapshot.getString( field: "lang").toString())
        }
    }
}
```

Рисунок 3.31 – Перелік коду : перевірка вибраної мови

Функція languageCheck () розташована в корені файлу MoreFragment.kt. Він перевіряє, яку мову вибрав користувач.

У випадку, якщо користувач вирішить не використовувати відповідну мову, Застосунок з’явиться англійською мовою.

```
@SuppressLint( ...value: "CommitPrefEdits")
private fun setLocale(lang: String) {
    val locale: Locale = Locale(lang)
    val res:Resources = resources
    val dm:DisplayMetrics = res.displayMetrics
    val conf:Configuration = res.configuration
    conf.locale = locale
    res.updateConfiguration(conf,dm)
    checkStateActivity()
}
```

Рисунок 3.32 – Перелік коду : встановлення правильної мови

Ми створюємо об'єкт класу Locale, який представляє певний географічний, політичний або культурний регіон. Доступ до ресурсів програми можна отримати за допомогою об’єкта класу Resources. Користувач використовує ресурси програми, тому що у нього є два різних каталога, які знаходяться в res / values / strings, і два окремих файли: strings.xml і strings.xml (uk-rUK), що містять рядки з текстом у кількох мови, strings.xml - англійська, strings.xml (uk-rUK) – українська. Ви можете використовувати клас DisplayMetrics, щоб отримати ширину та висоту екранів у пікселях.

Нарешті, дані оновлюються за допомогою функції updateConfiguration (conf, dm).

3.8. Модуль пошуку

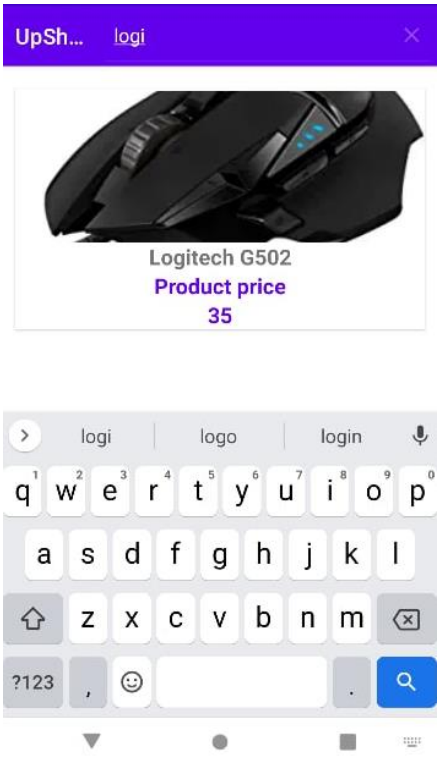

Рисунок 3.33 – Пошук у каталозі

Користувач може шукати продукти. Для цього і призначена панель пошуку. Після введення назви елемента відображається список з текстом, введеним користувачем. Після натискання на один з об’єктів користувач переходить на сторінку інформації про продукт, де він може його придбати або має можливість додати товар в кошик. Зацікавлена особа має можливість видалити написаний текст, а це означає, що список не відображатиметься. Цей фрагмент відповідає функціональним вимогам - 2.3.2 Пошук продукту.

```
override fun onCreateOptionsMenu(menu: Menu?): Boolean {
    inflater = menuInflater
    inflater.inflate(R.menu.menu_search,menu)
    var item: MenuItem = menu!!.findItem(R.id.search)
    var searchView: SearchView = item.actionView as SearchView
    searchView.isFocusable = true;
    searchView.setOnQueryTextListener(object: SearchView.OnQueryTextListener{
        override fun onQueryTextSubmit(query: String?): Boolean {
            Log.d( tag: "ChechOut",query.toString())
           // recyclerCart.visibility = View.VISIBLE
            return false
        }

        override fun onQueryTextChange(newText: String?): Boolean {
            Log.d( tag: "CheckOut",newText.toString())
            recyclerCart.visibility = View.VISIBLE
            adapter1.filter.filter(newText)
            return false
        }

    })
```

Рисунок 3.34 – Перелік коду : створення меню

```
    searchView.setOnQueryTextFocusChangeListener(object :View.OnFocusChangeListener{
        override fun onFocusChange(v: View?, hasFocus: Boolean) {
            if (hasFocus) {
                recyclerCart.visibility = View.GONE
            } else if (!hasFocus) {
                recyclerCart.visibility = View.GONE
            }
        }
    })
    return true
}
```

Рисунок 3.35 – Перелік коду : створення меню

Функція onCreateOptionsMenu розташована в кореневому каталозі файлу SearchFragment.kt. OnCreateOptionsMenu дозволяє створити меню. Це меню потрібне для введення запиту, а також пошуку та подання запиту постачальнику.

Завдяки SearchView.setOnQueryTextListener користувач може прослухати введений текст. Він виконує дві функції: після введення тексту та під час. У функції onQueryTextChange є адаптер, який стає видимим лише під час написання тексту.

```
private fun processSearch(query1: String?) {

    val docRef = fStore.collection( collectionPath: "products")
    docRef.get().addOnSuccessListener { documentSnapshot ->
        val city = documentSnapshot.toObjects<Products>()
        for(eachIndex in city.indices){
            if(eachIndex!=null){
                mUploads.add(city[eachIndex])
            }
        }
        if(mUploads.isEmpty()){
            Toast.makeText( context: this, text: "Empty",Toast.LENGTH_SHORT).show()
        }
        if(mUploads.isNotEmpty()){
            //Toast.makeText(this,"Empty",Toast.LENGTH_SHORT).show()
            adapter1 = FindItem(mUploads)
            recyclerCart.adapter = adapter1
            recyclerCart.visibility = View.GONE

        }

}
}
```

Рисунок 3.36 – Перелік коду : функція processSearch()

Функція processSearch () записує дані в масив, а потім вводить їх в адаптер з іменем FindItem.

```
private val exampleFilter: Filter = object : Filter() {
    override fun performFiltering(constraint: CharSequence): FilterResults {
        val filteredList: MutableList<Products> = ArrayList<Products>()
        if (constraint == null || constraint.length == 0) {
            filteredList.addAll(exampleListFull)
        } else {
            val filterPattern = constraint.toString().toLowerCase().trim { it <= ' ' }
            for (item in exampleListFull) {
                if (item.name!!.toLowerCase().contains(filterPattern)) {
                    filteredList.add(item)
                }
            }
        }
        val results = FilterResults()
        results.values = filteredList
        return results
    }

    override fun publishResults(constraint: CharSequence, results: FilterResults) {
        exampleList.clear()
        exampleList.addAll(results.values as List<Products>)
        notifyDataSetChanged()
    }
}
```

Рисунок 3.37 – Перелік коду : фільтрування та збереження даних пошуку

Поле exampleFilter, яке приймає значення з функції performFilter. Він перевіряє, чи введений користувачем текст схожий на назву продукту зі списку. Якщо у нас є подібні назви, ми зберігаємо їх у списку та показуємо.

3.9. Платіжний модуль

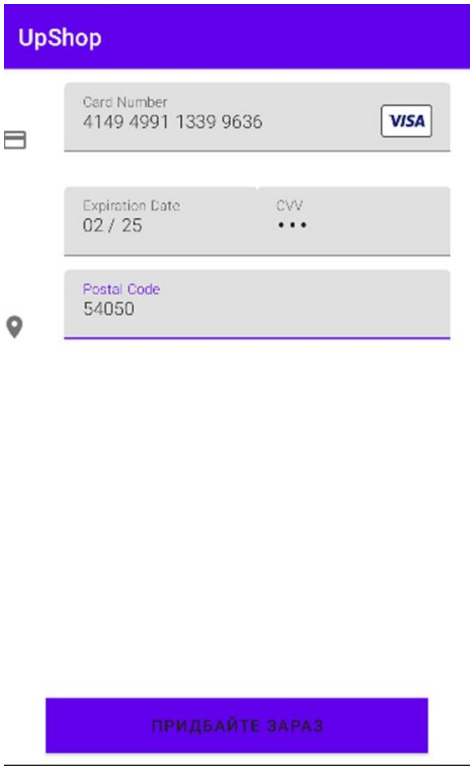

Рисунок 3.38 – Вид платежу

Користувач може вказати реквізити платіжної картки, форма перевіряє правильність введення номера картки та вказує тип картки, форма перевіряє правильність введених даних у полі терміну дії, у разі неправильно введених даних форма виконує не дозволяє натискати кнопку купити зараз. Форма виготовляється за допомогою «Android-card-form» .

3.10. Модуль адміністрування

Рисунок 3.39 – Подання адміністратора

Адміністратор бачить сторінку з 6 зображеннями, які представляють різні категорії товарів. Після натискання однієї з кнопок адміністратор переходить до активності додавання товарів. Він також має можливість контролювати замовлення клієнта. Після натискання на іконку «Перевірити новинки» адміністратор переходить на сторінку замовлення, де може побачити, які товари є в різних бронюваннях.

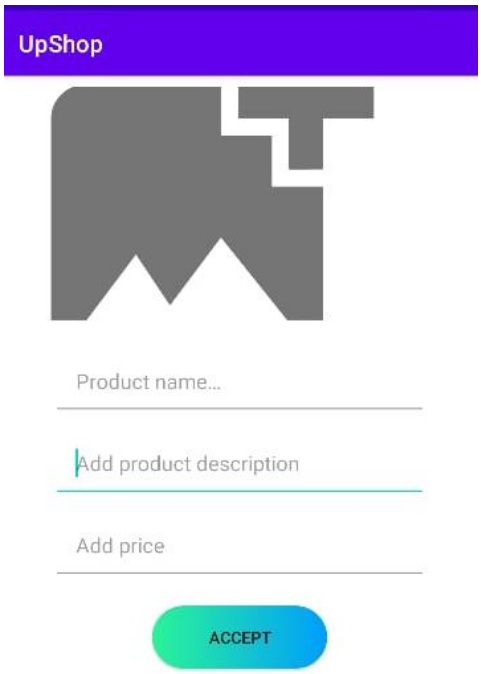

Рисунок 3.40 – Додавання продукту

На рисунку 3.40 показана форма додавання продуктів. Після натискання на картинку програма показує значки. Про процес додавання зображення на рисунку 3.41. Адміністратор повинен надати назву товару, опис і ціну. Грошові суми вказувати лише в цифрах. Після натискання кнопки «Прийняти» програма перевіряє правильність введених адміністратором даних, і після їх правильного введення продукт додається до бази даних. У разі неправильної інформації виводиться повідомлення про неправильні дані. Цей фрагмент відповідає функціональним вимогам - 1.1 Додавання, видалення та редагування продуктів.

```
productRandKey = saveCurrentDate + saveCurrentTime


val filePath:StorageReference = productImageRef.child( pathString: productRandKey + ".jpg")
uploadTask = filePath.putFile(imageUri)
uploadTask.addOnFailureListener{  it: Exception
    val message:String = it.toString()
    Toast.makeText( context: this, text: "Error: $message",Toast.LENGTH_SHORT).show()
    loadingBar.dismiss()
}
    .addOnSuccessListener {  it: UploadTask.TaskSnapshot!
        Toast.makeText( context: this, "Product Image uploaded successful",Toast.LENGTH_SHORT).show()
        var uriTask: Task<Uri> = uploadTask.continueWithTask(Continuation {  it: Task<UploadTask.TaskSnapshot!>

            if(!it.isSuccessful){
                throw it.exception!!
            }
            downloadImageUri = filePath.downloadUrl.toString()
            return@Continuation filePath.downloadUrl

        })
    }.addOnCompleteListener(OnCompleteListener {  it: Task<UploadTask.TaskSnapshot!>
        if(it.isSuccessful){
            downloadImageUri = it.result.toString()
            Toast.makeText( context: this, "Add new item",Toast.LENGTH_SHORT).show()
            saveProductInfoToDatabase()
        }
    })
```

Рисунок 3.41 – Перелік коду : додавання зображення до товару

Додавання зображення в базу даних можливо за допомогою класу Task і continueWithTask, що створює нове завдання. Він буде припинено в результаті зазначеного продовження, викликаного в основному потокі програми. Якщо нам вдасться виконати завдання, ми додаємо шлях до зображення до downloadImageUrl. Після перевірки всіх дійсних значень uploadTask.addOnCompleteListener відображає Toast про те, що продукт додано. Сценарій отримання зображень із галереї наведено на рисунку 3.42.

```
private fun openGallery(){
    val galleryIntent = Intent()
    galleryIntent.action = Intent.ACTION_GET_CONTENT
    galleryIntent.type = "image/*"
    startActivityForResult(galleryIntent, GalleryPick)
}
```

Рисунок 3.42 – Перелік коду : відтворення картинної галереї

3.11. Тести застосування

З метою перевірки якості системи та її роботи були проведені ручні тести функціонування мобільного додатка.

3.11.1. Реєстрація

Перший етап тестування – реєстрація в застосунку. Дані були правильно внесені в тестові поля, тому ускладнень не було. Потім для кожного з користувачів були створені облікові записи, про які вони були повідомлені.

Спочатку користувач вводить у форму дані: ім’я, прізвище, адресу електронної пошти, номер телефону та пароль. Після натискання кнопки «Зареєструватися» зацікавлена особа переходить на домашню сторінку, де можна побачити повідомлення про створення нового користувача.

3.11.2. Кошик

Після правильного входу в систему було перевірено функцію програми: додавання в кошик. Єдине, що ви помічаєте, це те, що грошові одиниці (долари) не відображаються. Проблема буде виправлена в наступній версії програми.

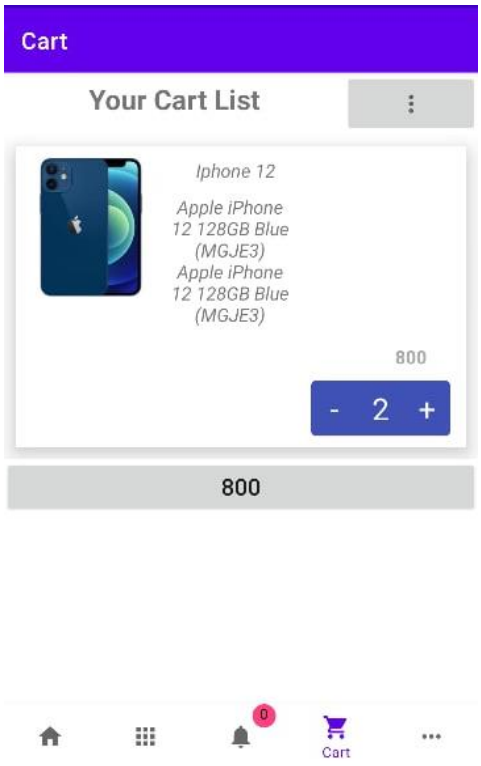

Рисунок 3.43 – Створення користувачем кошика для покупок

Після входу в систему користувач має можливість додавати продукти в кошик, змінювати кількість, видаляти їх і купувати, коли він натискає кнопку, яка переносить його до іншої діяльності.

Висновки до розділу 3

В даному розділі було розглянуто та проаналізовано засоби для розробки застосунків для Android. Під час створення застосунку було дотримано усіх вимог зазначених під час проектування та моделювання Застосунку. Було здійснено програмну реалізацію мобільного Застосунку інтернет-магазину з продажі електроніки.

Спеціальний розділ

ОХОРОНА ПРАЦІ

до кваліфікаційної роботи

на тему:

«МОБІЛЬНИЙ ЗАСТОСУНОК ДЛЯ ІНТЕРНЕТ-МАГАЗИНУ З ПРОДАЖУ ЕЛЕКТРОНІКИ»

Спеціальність 122 «Комп'ютерні науки»

122 – БКР – 402.21810221

Виконав студент 4-го курсу, групи 402

_____*І. С. Раздобудько*

(підпис, ініціали та прізвище)

«___» _____ 202_ р.

Консультант_____*ст. викладач*

(наук. ступінь, вчене звання)

_____*О. В. Макарова*

(підпис, ініціали та прізвище)

«___» _____ 202_ р.

Миколаїв – 2022

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

ЕП	–	Електронні прилади
ЕОМ	–	Електронна обчислювальна машина
КПО	–	Коефіцієнт природної освітленості
ПК	–	Персональний комп’ютер
ССБП	–	Система стандартів безпеки праці

4. ОХОРОНА ПРАЦІ

Вступ

У наш час, майже кожен бізнес використовує інформаційні технології, що передбачає роботу з різноманітними технічними інструментами. Персональний комп'ютер, наприклад, використовується для збору, розповсюдження та аналізу даних як всередині підприємств, так і між ними. Використання персонального комп'ютера значно спрощує різноманітні види діяльності. Однак слід пам'ятати, що вплив комп'ютера має як корисні, так і шкідливі наслідки.

На робочі обставини, зокрема у сфері IT-технологій, впливають різноманітні фактори виробничого середовища, найважливішим з яких є промислове освітлення. Цей аспект вплинув на мене найбільше, тому під час праці важливо виділити цей недолік.

Важливим обов'язком роботодавця є забезпечення комфортних і безпечних умов праці. Робоче середовище має безпосередній вплив на здоров'я та самопочуття робітників. Індивідуальні заходи з охорони праці, як показує практика на підприємствах, не дають бажаного ефекту, тому це питання потребує системної стратегії, в якій заходи з охорони праці здійснюються збалансовано та комплексно. З цією метою організації розробляють систему управління охороною праці, яка враховує її унікальні характеристики. Система управління охороною праці сприяє дотриманню стандартів зниження допустимих рівнів небезпечних і шкідливих змінних робочого середовища для працівників.

1. Опис можливих проблем, які можуть виникнути під час використання екранних пристроїв

Проблеми із зором та скелетно-м’язові травми є найбільш типовими симптомами, пов’язаними з роботою або процедурами ЕП. Напруга очей і головний біль - це дві візуальні змінні, які впливають на зорову продуктивність. Користувачі з уже наявними порушеннями зору можуть помітити підвищення їх видимості.

Ці симптоми можуть бути викликані якщо:

- залишатися в тому ж положенні та концентруватися на екрані ЕП протягом тривалого періоду часу;
- погане позиціонування ЕП;
- погана розбірливість екрана або вихідних документів;
- погане освітлення, включаючи відблиски та відображення;
- зображення на екрані, що блимає, мерехтить або тремтить.

Робоча діяльність, пов'язана з ЕП, була пов'язана з різноманітним дискомфортом у руках, кистях і плечах. Користувачі також можуть повідомляти про біль у руках, зап’ястях, передпліччях, шиї та нижній частині спини. Гострий штам може бути небезпечним.

Можуть виникати м’язова втома, ломота, біль, слабкість, хворобливість та/або набряк, які зникають після періоду спокою. Чим більше хронічних захворювань, тим більше симптомів. Тяжкі симптоми та порушення функції, що впливають на повсякденну діяльність

Подібні проблеми серед працівників клавіатур часто пов’язують з великими навантаженнями, коли терміни стислі. Це важливі фактори, які можуть зіграти певну роль.

Симптоми захворювання включають, але не обмежуються ними:

- позиції робочих станцій, які є фіксованими або стійкими;
- послідовні рухи;
- надмірна втома м’язів в результаті підвищеної напруги м’язів.

Випромінювання та його наслідки є менш відомою проблемою для здоров'я, пов'язаною з операціями ЕП. Зростає занепокоєння щодо випромінювання ЕП та їх потенційних наслідків для вагітних жінок. Згідно з дослідженнями низки країн, випромінювання електромагнітного випромінювання значно нижче нинішнього стандарту безпеки. Згідно з науковими дослідженнями, вагітним жінкам не потрібно припиняти використання ЕП. Жінкам, які вагітні або планують мати дітей і які стурбовані роботою з ЕП, слід надати можливість обговорити свої проблеми з кимось, хто достатньо поінформований з актуальною авторитетною науковою інформацією та порадами, щоб уникнути проблем, спричинених стресом та тривогою.

У дослідженнях стрес на роботі також був визначений як фактор ризику скарг на здоров'я користувачів ЕП. Погане самопочуття, нервозність, дратівливість і розлад травлення – одні з перших симптомів, які з’являються. У більшості випадків ці симптоми залишаються непоміченими і пов’язані з чимось іншим.

Деякі користувачі ЕП скаржилися на свербіж або почервоніння шкіри на обличчі та/або шиї. Ці скарги рідкісні, і мізерні дослідження показують, що вони пов’язані з умовами навколишнього середовища, такими як низька відносна вологість або статична електрика поблизу ЕП.

2. Нормативно-правові вимоги при роботі з екранними пристроями

Метою Закону про безпеку та гігієну праці 1994 року є забезпечення безпеки, здоров'я та благополуччя людей на роботі, захист інших від ризиків для їхньої безпеки чи здоров'я в результаті трудової діяльності людей, а також сприяти створенню професійного середовища. для людей на роботі, що відповідає їхнім фізіологічним та психологічним потребам.

2.1. Загальні обов'язки роботодавця

Роботодавці повинні максимально захищати здоров’я, безпеку та добробут своїх працівників. Це особливо актуально, коли йдеться про встановлення та обслуговування безпечної установки та робочої системи. Використання рослин і речовин, поводження з ними, зберігання та транспортування мають здійснюватися безпечним і здоровим способом.

Термін «завод» використовується в цьому законі для позначення будь-яких машин, обладнання, приладів, інструментів або компонентів, включаючи блоки відео-дисплеїв.

Крім того, законодавство зобов’язує роботодавців надавати працівникам відповідну інформацію, освіту, навчання та нагляд, щоб вони могли безпечно виконувати свою роботу. Іншим загальним обов’язком роботодавця, зазначеним у законодавстві, є підтримка безпечного робочого середовища та забезпечення наявності відповідних побутових засобів.

2.2. Відповідальність за формування політики безпеки та охорони праці

Роботодавець повинен розробити письмову заяву, в якій викладає свою загальну політику, організацію та положення щодо безпеки та здоров’я на робочому місці, оновлювати її шляхом внесення змін та надавати її працівникам.

Якщо роботодавець наймає на роботу 40 і більше осіб або за вказівкою генерального директора, на робочому місці має бути створена комісія з охорони праці. Основна відповідальність комітету полягає в тому, щоб відстежувати заходи, що застосовуються для захисту здоров’я та безпеки працівників на роботі, а також вивчати будь-які пов’язані питання, які виникають, наприклад скарги та випадки. Між роботодавцем і комітетом з питань безпеки та охорони праці завжди мають бути консультації та співпраця.

2.3. Обов'язки співробітників

Відповідно до закону, працівники зобов’язані проявляти розумну обережність, щоб уникнути заподіяння шкоди собі чи іншим особам у результаті своєї трудової діяльності. Вони також повинні працювати разом з роботодавцями та іншими особами для виконання законодавчих зобов’язань щодо безпеки та здоров’я на робочому місці. Співробітники також повинні дотримуватися цього закону, не втручаючись у те, що надається, і не зловживаючи ним, щоб захистити свою безпеку, здоров’я та благополуччя.

Якщо працівник подає скаргу щодо питання, яке, на його думку, є небезпечним, або якщо він чи вона виконує свою функцію як член комітету з безпеки та здоров’я, він чи вона захищені від дискримінації, травм або зміни його позиції на його чи її невигідність.

Про будь-який нещасний випадок, небезпечну подію, професійне отруєння або захворювання, які трапляються або можуть виникнути на роботі, необхідно повідомити місцеву службу безпеки та гігієни праці.

Кожен зареєстрований лікар або медичний працівник, який обслуговує або викликається до пацієнта, який, як він підозрює, страждає на одне з професійних отруєнь, перерахованих у Законі про фабрики та обладнання або названих в Законі про безпеку та гігієну праці (Декларація про професійне отруєння). Захворювання) Наказ 2000 р. також доповісти генеральному директору.

3. Методи профілактики

Різні компоненти, які сприяють ризику роботи ЕП, вимагають підходу до зниження ризику, який охоплює весь сценарій, який включає:

- дизайн робочого місця;
- фактори системи та обладнання;
- фактори, що впливають на робоче місце;
- характер і організація роботи;
- обслуговування обладнання та меблів ЕП;
- підбір персоналу та медичні огляди перед прийомом на роботу;
- навчання та інформація.

Конструкція робочого місця Робочі станції ЕП мають бути ергономічно побудовані з максимально можливою гнучкістю, щоб їх можна було налаштувати відповідно до потреб кожного окремого оператора.

3.1. Критерії вибору меблів для робочих місць

Щоб вибрати прийнятні меблі, слід керуватися наступними критеріями:

- завдання, виконані на робочому місці, наприклад:
    i. інформація обробляється;
    ii. автоматизована обробка текстів;
    iii. клавіатура;
    iv. типографіка;
    v. контрзаходи, такі як банківська справа;
    vi. комп'ютерне програмування.
- тривалість та інтенсивність завдання;
- обладнання, яке необхідно зберігати на робочому місці;
- фізичне середовище, в якому розташоване робоче місце;
- методи роботи обладнання;
- змінюється характер призначення робочої станції;
- незалежно від того, чи призначена робоча станція для одного користувача чи групи користувачів.

3.2. Робоча поверхня або робочий стіл

Розмір. Стіл або робоча поверхня повинні бути достатньо великими, щоб вмістити екран, клавіатуру, документи та інші підключені пристрої в гнучкому режимі. Співробітникам повинно бути достатньо місця, щоб вибрати зручну посаду.

Безпека. Гострі краї, кути, виступи або шорсткі поверхні робочого столу або робочої поверхні, особливо нижня сторона робочої поверхні, не повинні спричиняти травмування користувачів або пошкодження їхнього

одягу. Рухомі частини робочого місця або робочої поверхні не повинні створювати небезпеки.

Управління кабелями. Прилади для розміщення кабелів, необхідних для живлення, передачі даних і телефонних потреб робочої станції, а також для зберігання будь-якого додаткового кабелю повинні бути вбудовані в робочий стіл або робочу поверхню. Перемикачі, які часто використовуються, повинні бути легкодоступними. Обслуговування кабелю повинно бути включено в проект.

Приміщення для зберігання. На кожному робочому місці повинні бути передбачені складські приміщення для часто використовуваних предметів. Приміщення для зберігання повинні: а. бути стійким і не представляти небезпеки при повному навантаженні; б. бути сконструйованим так, щоб відкриватися, закриватися і блокуватися з нормального робочого положення; і с. мати принаймні одне відділення, що закривається, в якому оператор може зберігати особисті речі.

Приміщення для зберігання, розташоване під регульованою робочою поверхнею, повинні бути рухомими та висотою не вище 550 мм, щоб не перешкоджати повному налаштуванню робочої поверхні.

Регулювання висоти робочого столу або робочої поверхні. Де можливо, висоту слід налаштовувати відповідно до смаку кожного оператора. Наступні статистичні дані служать мірилом для рекомендацій щодо висоти столу або поверхні:

Обробка на робочій поверхні. Робоча поверхня повинна бути світло-коричневого нейтрального кольору з сатиновою або матовою обробкою. Робоча поверхня також повинна легко очищатися та оброблятися, щоб можна було писати на одному аркуші паперу без підкладки.

Робоче місце має бути міцним і міцним за своєю конструкцією. Під робочою поверхнею має бути достатньо місця, щоб ноги могли вільно рухатися без перешкод.

Простір для ніг не повинен заважати кабелями чи іншими пристроями.

Панно скромності. Для забезпечення конфіденційності оператора під робочою зоною має бути встановлена панель скромності. Панель скромності не повинна перешкоджати регулюванню висоти робочої поверхні.

3.3.Вимоги до крісла

Стільці повинні мати такі характеристики:

- стабільний, що дає оператору легку свободу рухів і зручне положення;
- регульована висота (газліфт для багатокористувацького робочого місця, гвинтове регулювання тільки для одного користувача) в діапазоні від 350 мм до 450 мм;
- спинка, яка регулюється як по висоті, так і по нахилу, щоб забезпечити адекватну підтримку спини, особливо в нижній частині спини (поперековий відділ);
- досить міцний чохол сидіння і «водоспад» перед;
- якщо є підлокітник, він не повинен перешкоджати роботі клавіатури;
- колеса або коліщатка (5-зіркова база) для стабільності та мобільності.

Кожен, хто вимагає підставку для ніг, має отримати її. Піддон сидіння повинен бути горизонтальним, де він закріплений. Якщо є нахил, рекомендується максимум 10 градусів вперед і 5 градусів назад. Поверніть сидіння навколо вертикальної осі, яка проходить через плоску частину сидіння.

3.4. Налаштування меблів та обладнання ЕП

Щоб уникнути неприємної пози під час безперервної роботи, висоту сидіння, положення клавіатури, термінала дисплея, висоту робочої поверхні та інші фактори слід регулювати разом. Для цього слід мати на увазі наступне:

- оператор повинен мати можливість сидіти в кріслі з належною опорою спини на спинку, а його підошви взуття торкаються підлоги. При необхідності слід передбачити неслизьку підставку для ніг відповідного розміру;
- висоту сидіння слід відрегулювати, щоб запобігти надмірному тиску під стегном оператора. Бажано залишити достатньо місця між нижньою стороною його ноги та сидінням, щоб його пальці могли плавно ковзати;
- з опущеними надпліччями практично перпендикулярно та під кутом 90 градусів або під більш широким кутом між ними, пальці повинні мати можливість досягти клавіатури природним чином;
- верхній край екрана повинен бути нижче рівня очей оператора. Відстань для огляду повинна бути не менше 400 мм;
- екран дисплея, клавіатура та/або документ повинні бути встановлені в межах достатнього діапазону зору, щоб уникнути різких перепадів відстані від очей оператора.

Вимоги до утримувача документів, а також належне оформлення та застосування такого обладнання визначаються роботою. Щоб зменшити втому м’язів очей і шиї, тримачі для документів необхідні. На малюнку 1 показано три приклади відповідних позицій власника документів для трьох окремих робіт.

3.5. Вологість і температура в навколишньому середовищі

- ЕП та супутнє до них обладнання виробляють тепло. Це слід при розміщенні ЕП. У протидії вентиляції необхідна достатня та кондиціонування повітря;
- повітряні потоки від вентиляторів ЕП слід спрямувати на подалі від оператора, тому надто силовий рух повітря може викликати подразнення поверхні;
- на етапі придбання слід розглянути блоки ЕП з низькими тепловими викидами та супутнє обладнання. Постачальники обладнання можуть додавати цю інформацію;
- температура навколишнього повітря повинна бути від 23 до 27 градусів за Цельсієм, максимальна відносна вологість – 75%.

3.6. Рівень шуму в навколишньому середовищі

- робоча зона для ЕП має бути дещо тихою, з невеликою кількістю відволікаючих дій та шуму;
- шум від принтерів та іншого офісного обладнання можна пом’якшити за допомогою ізоляції, капюшона та екранування. Найкраще уникати використання незакритих матричних принтерів поблизу операторів;
- шум від вентиляторів охолодження, блоків живлення та клавіатури слід звести до мінімуму, і це має бути головним пріоритетом під час покупок. Якщо обладнання видає занадто багато шуму, його потрібно негайно обслуговувати;
- для роботи ЕП рекомендується рівень шуму навколишнього середовища 40-60 дБ (А).

4. Оцінка природнього освітлення у виробничому приміщенні

Природне освітлення – освітлення приміщень денним світлом, що потрапляє через світлові отвори (вікна) в зовнішніх захищаючих конструкціях приміщення. Природне освітлення характеризується тим, що змінюється в широких межах залежно від часу дня, пори року і ряду інших чинників.

Перевірочний розрахунок виконується в такій послідовності:

Нормативне значення коефіцієнта природного освітлення для III поясу світлового клімату e_H^{III} *, %.*

Визначається відповідно до СНиП II - 4 - 79 [2, таблиця 3.1]. Для зорових робіт середньої точності при найменшому розмірі об'єкта розпізнавання 0,5-1 мм при боковому освітленні (так званий IV розряд зорової роботи):

$$e_H^{III} = 1{,}0\%$$

Коефіцієнт світлового клімату m.

Для Миколаївської області, що належить до IV поясу світлового клімату відповідно до рекомендацій [2]:

$$m = 0{,}9.$$

Коефіцієнт сонячності клімату c.

Для світлових отворів (вікон) в зовнішніх стінах будівель, розташованих у IV поясі світлового клімату та зорієнтованих по азимуту в діапазоні 136... 225 градусів відповідно до рекомендацій [2, табл. 3.3]:

$$c = 0{,}7.$$

Нормоване значення коефіцієнта природного освітлення для розрахункових умов $e_н$*, %:*

$$e_H = e_H^{III} \cdot m \cdot c = 1{,}0 \cdot 0{,}9 \cdot 0{,}7 = 0{,}63\%.$$

Коефіцієнт запасу, що використовується при розрахунку природного освітлення к3.

Відповідно до рекомендацій [2]: $к_З$ = 1,3...1,5.

Прийнято $к_З$ = 2,0.

Відношення довжини приміщення a до його ширини b, a/b:

$a/b = 6/6 = 1.$

Відношення ширини приміщення b до відстані від верхньої кромки вікна до робочої поверхні h, b/h:

$b/h = 6/2,3 = 2,61.$

Світлова характеристика вікна ηв.

Визначається відповідно до рекомендації [2, табл. 3.4] $ηв = f(a/b, b/h)$ при a/b = 1 і b/h= 2,61

$$η_в = 13.$$

Коефіцієнт світлопропускання матеріалу $τ_1$.

Визначається згідно рекомендацій [2, табл. 3.6] відповідно до подвійного склопакета металопластикових вікон: $τ_1$=0,86.

Коефіцієнт, що враховує втрати світла у віконній рамі $τ_2$.

Визначається згідно рекомендацій [2, табл. 3.6]. Для рам металопластикових вікон: $τ_2$ = 0,76.

4.2.11. Коефіцієнт, що враховує втрати світла у несучих конструкціях $τ_3$.

При боковому освітленні згідно рекомендацій [2]: $τ_3$=1.

Коефіцієнт, що визначає втрату світла у сонцезахисних конструкціях $τ_4$.

Для внутрішніх регульованих жалюзі з рекомендації [2]: $τ_4$=1.

4.2.13 Загальний коефіцієнт світлопропускання $τ_{заг}$

$$τ_{заг} = τ_1 \cdot τ_2 \cdot τ_3 \cdot τ_4 = 0,86 \cdot 0,76 \cdot 1 \cdot 1 = 0,65$$

Коефіцієнт відбиття внутрішніх поверхонь приміщення: ρстелі, ρстін, ρпідлоги, %

Дані показники визначаються відповідно до [2, табл. 3.8-3.10] для:

- для гладкої поверхні білої напівматової стелі $ρ_{стелі}$ = 82%;

- для стін, обклеєних шпалерами світлого тону, $\rho_{стін}$ =40%;
- для підлоги $\rho_{підлоги}$ = 40%.

Площі внутрішніх поверхонь: $S_{стелі}$, $S_{стін}$, $S_{підлоги}$, м2

$$S_{стелі} = a \cdot b = 6 \cdot 6 = 36 \text{ м}^2$$

$$S_{стін} = 2 \cdot (a+b) \cdot H = 2 \cdot (6+6) \cdot 3{,}5 = 84 \text{ м}^2$$

$$S_{підлоги} = 36 \text{ м}^2$$

Середнє значення коефіцієнта відбиття $\rho_{сер}$.

$$\rho_{сер} = \frac{\rho_{стелі} \cdot S_{стелі} + \rho_{стін} \cdot S_{стін} + \rho_{підлоги} \cdot S_{підлоги}}{(S_{стелі} + S_{стін} + S_{підлоги}) \cdot 100} = \frac{82 \cdot 36 + 40 \cdot 84 + 40 \cdot 36}{(36 + 84 + 36) \cdot 100} = 0.497$$

4.2.17 Співвідношення, що характеризують геометрію приміщення a/b, b/h, l/b

$$a/b = 1;\ b/h = 2{,}61;\ 1/b = 0{,}17.$$

Коефіцієнт, що враховує підсилення природного освітлення за рахунок світла, що відбивається від внутрішніх поверхонь приміщення r_1.

Відповідно до рекомендацій [2, табл. 3.7] $r_1 = f(\rho_{сер}, a/b, b/h, 1/b)$. При $\rho_{сер}$ = 0,482, a/b =1; b/h = 2,61; 1/b = 0,17.

$$r_1 = 2.$$

Відношення відстані до протилежної будівлі D до висоти карнизу протилежної будівлі над підвіконням H', D/H':

$$D/H' = 16/6 = 2{,}67.$$

Коефіцієнт, що враховує вплив протилежної будівлі на освітленість у приміщенні, $к_{буд}$.

Відповідно до [2, табл. 3.5] $к_{буд} = f\,(D/H')$ =2,67

$$к_{буд} = 1{,}0.$$

Площа вікон, що необхідна для забезпечення нормованого коефіцієнта освітлення у розрахунковому приміщенні SB, м²:

$$S_B = \frac{e_н \cdot к_з \cdot \eta_в \cdot S_{підлоги} \cdot K_{буд}}{\tau_{заг} \cdot r_1 \cdot 100} = \frac{0{,}63 \cdot 2{,}0 \cdot 13 \cdot 20 \cdot 1{,}0}{0{,}65 \cdot 2 \cdot 100} = 2{,}52 \text{ м}^2$$

В приміщенні розташовано 2 вікна розмірами d x c = 1,5 x 2,2 = 3,3 м².

Їх загальна площа складає $S_{заг}$= 6,6 м².

Висновки до розділу

Під час виконання спеціального компонента з охорони праці оцінювалися проблеми, які можуть виникнути у робітників під час праці з екранними пристроями, а також методи рішення та протидіям цих проблем.

Оскільки практично в кожній організації є співробітники, які працюють з електронним обладнанням, правила, викладені в нормативних актах, діють і сьогодні. Робота за персональним комп’ютером швидко виснажує людину і викликає проблеми з опорно-руховим апаратом, очами, неврологічними, а в деяких випадках і з психічним здоров’ям.

Сьогодні численні небезпечні ситуації, такі як шум, вібрація та інші фактори, загострюють процес людської діяльності.

Під час життєдіяльності людини необхідно дотримуватися вимог і правил охорони праці. Співробітники та власники бізнесу, які дотримуються правил, зменшать негативний вплив на здоров’я людини.

Висновки

Метою роботи було розробка та впровадження мобільного застосунку – клієнта інтернет-магазину. Ця програма дозволяє входити та реєструвати користувачів, переглядати товари, шукати їх, змінювати персональні дані, здійснювати онлайн-платежі, розміщувати товари в кошику та видаляти їх. Програма відповідала всім функціональним і нефункціональним вимогам, зазначеним у проекті.

Застосунок реалізовано за допомогою мови програмування Kotlin, що полегшує написання коду. У системі використовується новий тип баз даних - NoSql. Платформою, яка використовує базу даних такого типу, є Firebase, яка дозволяє швидко створити базу даних. Завдяки використанню сучасних технологій дані програми є безпечними. При написанні коду програми увагу було приділено використанню передових практик, завдяки яким Застосунок можна розширити новими функціями.

Застосунок можна розробити з новими функціями або вдосконалити наявні. Непогано додати функцію користувача – повернення товару в разі технічної несправності товару або іншого випадку. Для адміністратора ви також можете додати функцію тестових покупок, щоб перевірити правильність роботи програми. Створений клієнт інтернет-магазину також можна розширити версією iOS.

Робота над проектом дозволила використати набуті під час навчання навички для створення повністю функціональної системи та спонукала до вивчення нових технологій, таких як типи баз даних NoSql.

Список використаної літератури

[1]. Зеркалов В.Д. З-57 Охорона праці в галузі: Загальні вимоги. Навчальний посібник, 2011. - 551 с.

[2]. Піменов Сергій. Язык программирования Kotlin. Навчальний посібник, 2019. - 304 с.

[3]. Девід Гринхол. Kotlin. Программирование для профессионалов. Навчальний посібник, 2020. - 251 с.

[4]. Jemerov D., Isakova S.: Kotlin in Action. Manning, 2017.

[5]. Moskala M., Wojda I.: Android Development with Kotlin, BIRMINGHAM - MUMBAI 2017.

[6]. W. Frank Ableson, King C., Ortiz E.: Android in Action, Third Edition, Manning, 2011.

[7]. Laurence M.: The Definitive Guide to Firebase, Laurence Moroney, 2017.

[8]. Stonehem B.: Google Android Firebase: Learning the Basics, First Rank Publishing, 2016.

[9]. Verhagen T., van Dolen W.: The influence of online store beliefs on consumer online impulse buying: A model and empirical application, 2011.

[10]. Google Play, *https://play.google.com/store?hl=en&gl=US, [*Дата звернення: *15.04.2022]*

[11]. Google Play Rozetka, *https://play.google.com/store/apps/details?id=ua.com.rozetka.shop,[* Дата звернення: *21.04.2022]*

[12]. Firebase, *https://firebase.google.com/, [*Дата звернення: *01.05.2022]*

[13]. GitHub, *https://github.com/,* [Дата звернення: 02.05.2022]

[14]. Статистика проданих телефонів у 2020 році,

*https://gs.statcounter.com/os-market-share/mobile/worldwide, [*Дата звернення: *26.04.2022]*

*[15]. Статистика продажів у США, https://www.forrester.com/blogs/us-smartphone-and-tablet-online-retail-forecast-update/ , [*Дата звернення: *26.04.2022]*

[16]. Мобільний Застосунок для аналізу доходів, *https://www.forbes.com/sites/tristanlouis/2013/08/10/how-much-do-average-apps-make/?sh=3abcdedc46c4, [*Дата звернення: *10.05.2022]*

[17]. AVR PROGRAMMING 01: INTRODUCTION URL: https://hackaday.com/2010/10/23/avr-programming-introduction/ [Дата звернення: 14.05.2021].

[18]. ДБН В.2.5-28:2018. Природне і штучне освітлення. Державні будівельні норми України : веб-сайт. URL: http://dbn.co.ua/load/normativy/dbn/dbn_v_2_5_28/1-1-0-1188. [Дата звернення: 25.05.2022]