МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ ЕЛЕКТРОНІКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему: «Інформаційна система організації замовлення бутильованої питної води»

за спеціальністю 122 «Комп'ютерні науки»,
освітньо-професійна програма «Інформаційні технології проектування»

Виконавець роботи: студент групи ІТ.м-01 Андрусишин Іван Костянтинович

Кваліфікаційну роботу захищено на засіданні ЕК з оцінкою _____ «___» грудня 2021 р.

Науковий керівник _____ к.т.н., доц., Парфененко Ю.В.
(підпис)

Голова комісії _____ Шифрін Д.М.
(підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент _____
(підпис)

Суми-2021

Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра інформаційних технологій
Спеціальність 122 «Комп'ютерні науки»
Освітньо-професійна програма «Інформаційні технології проектування»

ЗАТВЕРДЖУЮ
Зав. кафедри ІТ

_____ В. В. Шендрик
«___» _____ 2021 р.

ЗАВДАННЯ

на кваліфікаційну роботу магістра студентові

Андрусишин Іван Костянтинович
(прізвище, ім'я, по батькові)

1 Тема проекту ***Інформаційна система організації замовлення бутильованої питної води***

затверджена наказом по університету від «29 » 10 2021 р. № 0787-VI

2 Термін здачі студентом закінченого проекту «10» грудня 2021 р.
3 Вхідні дані до проекту перелік вимог для розробки інформаційної системи організації замовлення питної бутильованої води

4 Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити) аналіз предметної області, постановка задачі та методи дослідження, проектування інформаційної системи організації замовлення питної бутильованої води, розробка інформаційної системи

5 Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Мета та задачі кваліфікаційної роботи магістра, актуальність розробки, порівняння мобільних додатків для замовлення питної води, функціональні вимоги до інформаційної системи, контекстна діаграма інформаційної системи, діаграма декомпозиції, діаграма варіантів використання модель даних, засоби реалізації, архітектура інформаційної системи, демонстрація мобільного додатку, висновок

6. Консультанти випускної роботи із зазначенням розділів, що їх стосуються:

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

Дата видачі завдання _____.

Керівник _____
(підпис)

Завдання прийняв до виконання _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів випускної проекту	Термін виконання етапів проекту	Примітка
1	Аналіз предметної області	01.09.21-15.09.21	
2	Підготовка специфікації	16.09.21-25.09.21	
3	Постановка задачі	26.09.21-30.09.21	
4	Створення документації	01.10.21-05.12.21	
5	Вибір засобів реалізації	06.10.21-10.10.21	
6	Розробка дизайну	11.10.21-20.10.21	
7	Розробка бази даних	21.10.21-30.10.21	
8	Реалізація макетів	01.10.21-10.11.21	
9	Написання модулів системи	10.11.21-30.11.21	
10	Тестування інформаційної системи	01.12.21-05.12.21	

Магістрант _____ Андрусишин І.К..

Керівник роботи _____ к.т.н., доц. Парфененко Ю.В.

РЕФЕРАТ

Тема кваліфікаційної роботи магістра «Інформаційна система організації замовлення бутильованої питної води».

Пояснювальна записка складається зі вступу, 4 розділів, висновків, списку використаних джерел із 34 найменувань, 2 додатків.

Кваліфікаційну роботу магістра присвячено розробці інформаційної системи організації замовлення бутильованої питної води.

У першому розділі проведено огляд останніх публікацій за тематикою роботи та огляд етапів розробки мобільних додатків. Також було проаналізовано аналоги інформаційної системи, що знаходяться у відкритому доступі та визначено їх функціонал.

У другому розділі поставлено задачі для розробки інофрмаційної системи, визначено функціональні вимоги та список користувачів. Окрім цього, було проаналізовано технології та засоби реалізації, визначено їх переваги для використання.

У третьому розділі проведено структурно-функціональне моделювання, змодельовано варіанти використання інформаційної системи користувачами та спроектовано базу даних.

У четвертому розділі наведено архітектуру інформаційної системи, та показано реалізацію інформаційної системи згідно етапу проектування з прикладами роботи.

Ключові слова: мобільний додаток, інформаційна система, android, доставка води, розробка.

ЗМІСТ

ВСТУП

Вживання людиною води щодня є обов'язковою умовою існування. Але актуальним питання залишається те, яка вода є корисною, а яка може причинити шкоди організму. Але у зв'язку з тим, що вода з водопроводу в основному не відповідає високій якості, люди нерідко віддають перевагу бутильованій воді, безпеку якої гарантує виробник.

Можливість покупок в Інтернеті полегшила ситуацію для всіх типів покупців. Магазини, ресторани та інші постачальники послуг і продуктів надали клієнтам доступ до своїх продуктів онлайн. Інформаційні технології надають доступ до товарів у будь-який час та у будь-якому місці. А для бізнесу перевагами є заохочення до покупки клієнтами, їх утримання та організація роботи компанії. Щоб користуватися такими перевагами та можливостями, необхідно мати онлайн-додаток для доставки.

Сьогодні мобільний телефон став повноцінною частиною кожного бізнесу. Галузь мобільних додатків для бізнесу пройшла довгий шлях, і користувачі мобільних пристроїв тепер можуть виконувати складні завдання, такі як оплачувати рахунки, реєструватися на рейси, стежити за щоденними завданнями тощо, просто використовуючи свої мобільні телефони. Бізнес-додатки також мають багато інших переваг для користувачів, постійно покращуючи їх продуктивність та економлячи час і гроші. Від співробітника початкового рівня до вищого керівництва — сьогодні кожен використовує мобільний бізнес-додаток, щоб залишатися продуктивним на ходу.

Об’єкт дослідження – інформаційне забезпечення процесу доставки бутильованої питної води.

Предмет дослідження – технологія розробки інформаційної системи для організації замовлення бутильованої питної води.

Метою даної роботи є розробка інформаційної системи для організації замовлення питної води у вигляді мобільного додатка.

Для досягнення мети необхідно виконати наступні задачі:

– провести аналіз предметної області;

– провести аналіз сучасних технологій за тематикою проекту;

– ознайомитись з аналогами додатків із даної теми, виділити їх основні переваги та недоліки;

– обрати технології для розробки даного продукту;

– розробити та реалізувати модель та структуру мобільного додатку;

– розробити функціонал для організації замовлення питної води;

– провести тестування системи.

Практичне значення розроблюваного додатку полягає в скороченні часу користувача чи менеджера компанії під час оформлення замовлення на покупку питної води і супутніх товарів та визначення оптимального маршруту доставки по точкам адрес клієнтів для кур’єра.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд останніх публікацій

Мобільні телефони стали невід'ємною частиною життя людства. Різні програми, що інтегровані в мобільні пристрої, стали дуже корисними в сучасному житті. Сукупний прогрес існуючих мобільних технологій, наявність високошвидкісного Інтернету та зручний інтерфейс у пристроях – все це призвело до появи мобільних додатків різного призначення [1].

Доступність мобільних додатків на сьогоднішній день зросла настільки стрімко, що це спричинило зміну в сприйнятті і використанні обчислювальних робіт. В даний момент мобільні додатки в основному використовуються в наступних сферах [2]:

– соціальні мережі;
– спілкування;
– покупки;
– замовлення товарів;
– банкінг;
– освіта;
– бізнес та інші.

Завдяки додаткам мобільний пристрій для великої кількості людей став невід’ємною частиною. Наприклад, телефон допомагає у забезпеченні організованого життя такими програмами, як контакти, нагадування, будильники, списки справ та інші, які можна підлаштувати на індивідуальні потреби та вимоги, що зробить життя комфортним, простішим та продуктивнішим.

У секторі банківської справи були виконані зусилля для пришвидшення та полегшення грошових операцій [3-4]. Щодо фінансових операцій, за

допомогою додатків можна швидко та зручно оплачувати товари та послуги, не користуючись послугами банку.

Останнім часом мобільні програми змінили соціальні медіа. Це вже не тільки засіб підтримки зв’язку, але й платформа для бізнесу. Це стало можливим завдяки появі додатків у соцмережах, які можна вважати одними з найефективніших та найдешевших інструментів для бізнесу [5-6].

Поява та доступність таких мобільних додатків, як ігри, відео- та музичні додатки для телефонів користувачів довели, що програми мають значення не тільки для спілкування та бізнесу, але також і для відпочинку та розваг людей.

Безсумнівно, що мобільні програми будуть існувати ще довгий час. Хоча технології будуть змінюватися, як і способи створення самих додатків, так чи інакше їх доведеться використовувати хоча-б для зв’язку між собою, здійснення покупок чи оплати рахунків [7].

Сучасний прискорений спосіб життя та інтенсивне використання технологій підштовхують клієнтів купувати через мобільні додатки [8-9]. Люди шукають нові альтернативи, щоб полегшити повсякденні завдання та адаптувати їх до свого способу життя [10-11]. Зокрема, логістичні служби все частіше пропонують інноваційні рішення, починаючи від доставки додому, закінчуючи постачальниками логістичних послуг, які розміщують пакети електронної комерції в пунктах самостійного збору [12]. Служби доставки додому зручні для онлайн-покупців [13] і є невід'ємною частиною міських логістичних послуг [14]. Крім того, завдяки додаткам для мобільних телефонів послуги доставки до дому стали ще важливішими, оскільки клієнти бажають отримувати продукти в потрібний час, в потрібному місці, у потрібній кількості та в потрібному стані [15-16].

Феномен служби доставки було визначено як «послуга доставки, яку пропонує магазин для доставки своїх товарів до дому покупця» [17]. Серед послуг онлайн-доставки доставка питної води в порівнянні з їжею не є швидкозростаючою через невисокий попит [18-19]. Доставка до дому

забезпечує додаткову вартість для всіх учасників системи, яка, хоча й ускладнює процес розподілу, створює конкурентну перевагу, краще задовольнивши клієнтів [20-21].

Дослідження [8] надає результати, як споживачі приймають технології безпосередніх мобільних платежів, яка дозволяє їм оплачувати покупки в реальному магазині за допомогою своїх смартфонів. В роботі [13] наведено, що онлайн-покупці наголошують на безпеці особистої інформації та механізмах торгівлі цією інформацією. У статті [17] досліджується організація послуг доставки їжі з точки зору користувача мобільного додатка через зручність, яку надають програми. Стаття [20] вивчає ефективні IT-технології маршрутизації транспортних засобів, які можна застосовувати в мобільних додатках.

З огляду на проведений аналіз мобільний додаток для організації замовлення доставки є досить актуальним на сьогодні.

1.2 Огляд етапів розробки мобільних додатків

Задача доставки питної води з точки зору IT фахівця полягає в створенні інформаційної системи, яка буде в будь-який час доступна як і для клієнтів, так і для персоналу. Через те, що мобільний телефон завжди під рукою, зручно використовувати саме мобільний додаток.

Ефективний процес розробки мобільного додатку охоплює шість ключових етапів [22]:

- Стратегія розробки;
- Аналіз і планування;
- Дизайн UI / UX;
- Розробка додатку;
- Тестування;

– Розгортання та підтримка.

Незалежно від розміру та масштабів проекту, виконання цього процесу розробки зробить ініціативу з розробки мобільних додатків для підприємства успішною.

Перший етап процесу розробки мобільного додатка — це визначення стратегії розвитку ідеї в успішний додаток. Оскільки цілі одного додатка можуть відрізнятися від іншого, стратегія мобільності все одно має вплив на стратегію мобільності, яку потрібно вирішити під час процесу розробки.

На цьому етапі визначаються типи користувачів програми, досліджуються аналоги, встановлюються цілі, завдання та вибір платформи для додатку.

На етапі аналізу і планування ідея майбутньої програми починає створюватися і переростає в реальний проект. Етап починається з визначення варіантів використання та фіксації детальних функціональних вимог.

Визначивши вимоги до програми, потрібно підготувати дорожню карту продукту. Це включає визначення пріоритетності вимог мобільного додатка та їх групування за етапами доставки.

Етап планування включає в себе визначення навичок, необхідних для початку розробки програми. Наприклад, для мобільних операційних систем iOS та Android потрібно використовувати окремі стеки технологій розробки. Але якщо основною метою є створення мобільного додатку як і для операційної системи iOS, так і для Android, команда розробників мобільних пристроїв має включати розробників iOS і Android.

Метою дизайну програми є забезпечення зручної та швидкої роботи користувача з красивим інтерфейсом.

Успіх мобільного додатка визначається на основі того, наскільки добре користувачі використовують усі його функції. Мета дизайну UI/UX для мобільних додатків полягає в створенні позитивного досвіду користувача, що робить програму інтерактивною, інтуїтивно зрозумілою та зручною для користувачів. У той час як відшліфований дизайн інтерфейсу користувача

допоможе з раннього впровадження, розроблюваний додаток повинен надавати інтуїтивно зрозумілий та простий користувальницький досвід, щоб підтримувати зацікавленість користувачів програми.

Дизайнери мобільних додатків часто починають дизайн додатків із ескізів на папері. Каркаси — це цифрова форма ескізів. Каркаси зосереджені більш на естетиці та досвіді користувача, а не на колірних схемах і стилях. Створення каркасів — це швидкий і економічно ефективний підхід до розробки макетів додатків і повторення їх у процесі реалізації дизайну. При створенні каркасів слід враховувати специфічний дизайн пристрою.

Посібники зі стилів сприяють стратегії дизайну програми. Створення керівництва зі стилю на початку процесу розробки мобільного додатка покращує продуктивність розробників мобільних додатків. У той же час дотримання посібника зі стилю допоможе зберегти вигляд додатка послідовним.

Макети або високоякісні дизайни — це остаточне відтворення візуального дизайну розроблюваного додатка. Макети створюються шляхом застосування посібника зі стилю до каркасів програми.

У той час, як макети відображають функціональні можливості мобільного додатка за допомогою статичного дизайну, вони можуть перетворитися на прототипи. Прототипи дуже корисні для моделювання роботи користувача та робочих процесів програми, які очікуються від готового продукту. Хоча розробка прототипів може зайняти багато часу, вони пропонують тестування дизайну та функціональності додатка на ранній стадії. Часто прототипи допомагають визначити модифікації функціональності програми.

Планування залишається невід’ємною частиною цього етапу процесу розробки мобільного додатка. Перш ніж розпочати фактичні зусилля з розробки/програмування, потрібно:

– визначити технічну архітектуру;
– вибрати стек технологій;

– визначити етапи розвитку.

Стандартний проект мобільного додатка складається з трьох обов’язкових частин: back-end, API та інтерфейсу мобільного додатка.

Back-end частина містить базу даних і об’єкти на стороні сервера, необхідні для підтримки функцій мобільного додатка.

Інтерфейс прикладного програмування (API) — це спосіб зв’язку програми з внутрішнім сервером або базою даних. Часто мобільні програми складаються з інтерактивного користувальницького інтерфейсу, який використовує серверну частину та API для керування даними. У деяких випадках, коли програмі потрібно дозволити користувачам працювати без Інтернету, програма може містити локальне сховище даних.

Для серверної частини можна використовувати майже будь-яку мову веб-програмування та бази даних. Для нативних мобільних додатків потрібно вибрати технологічний стек, необхідний для кожної платформи мобільної ОС. Програми iOS можна розробляти за допомогою мови програмування Objective-C або Swift. Програми для Android в основному створюються за допомогою Java або Kotlin.

Після завершення кожного етапу розробки додаток переходить на стадію тестування програми для перевірки.

Виконання тестування забезпечення якості (QA) при розробці мобільного додатка робить програму стабільною, зручною та безпечною. Окрім цього, тестування допомагає визначити можливі помилки у роботі програми, з їх подальшим виправленням.

Щоб випустити вбудований мобільний додаток, потрібно надіслати його в магазини додатків, Apple App Store для додатків iOS і Google Play для додатків Android. Однак перед запуском мобільної програми вам знадобиться обліковий запис в Apple App Store і Google Play Store .

Для випуску програми в магазині додатків потрібно підготувати метадані, зокрема назва додатку, його опис, категорія, ключові слова, та скріншоти робочої програми.

Після відправки в Apple App Store додатку iOS проходять процес перевірки, який може тривати від кількох днів до кількох тижнів залежно від якості програми та того, наскільки вона відповідає інструкціям Apple із розробки iOS. Якщо програма вимагає від користувачів авторизацію, тоді потрібно надати Apple обліковий запис користувача для тестування.

У додатках Android немає процесу перевірки, і вони стають доступними в магазині додатків протягом кількох годин після подання.

1.3 Аналіз мобільних додатків доставки води

Аби визначити переваги, які будуть присутні в розроблюваному додатку, потрібно провести огляд існуючих програмних продуктів-аналогів, які використовуються для вирішення подібних задач. Але так як основний функціонал продукту, а саме організація доставки питної води буде застосовуватися компанією, аналоги не знаходяться у відкритому доступі. Окрім цього, для кожної компанії треба власний додаток, і аналоги не підійдуть. В зв’язку з цим було прийнято рішення проаналізувати програми за тематикою роботи, які мають відкритий доступ до більш меншого функціоналу.

Серед списку додатків у Play Market за пошуком запитом “Доставка води” (рис. 1.1) в якості дослідження було розглянуто наступні додатки для організації замовлення питної води: Люкс Вода, ЯП Вода та Big Bottle.

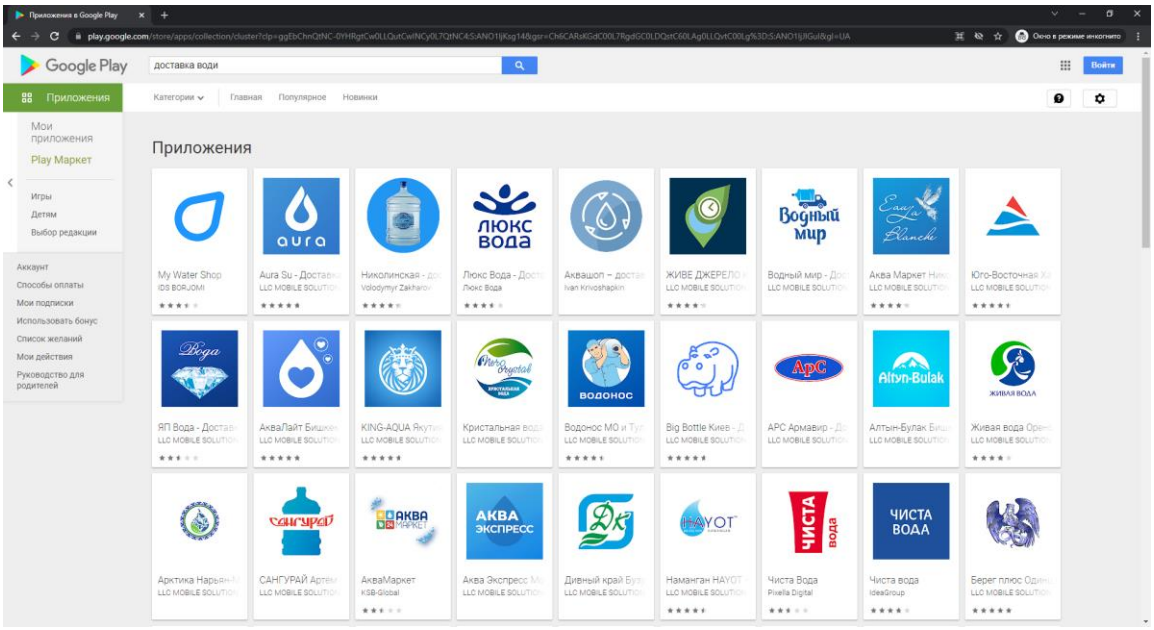


Рисунок 1.1 – Аналоги

На головній сторінці додатку розташовані кнопки для швидкого переходу за категоріями, а саме вхід до особового кабінету, замовлення води, кулери та помпи, також акції та супутня продукція (рис. 1.2).

Рисунок 1.2 – Головний екран додатку Люкс Вода

Бокове меню містить посилання на головну сторінку, на карту, бонуси, промокоди, на оформлення замовлення, кошик, історію замовлень, розташування магазинів, акції, новини, сповіщення та інше.

Вибір замовлення відбувається на окремій сторінці, на якій розміщено меню з категорією товару та слайдер з вибором товару для води чи список для інших категорій (рис. 1.3).

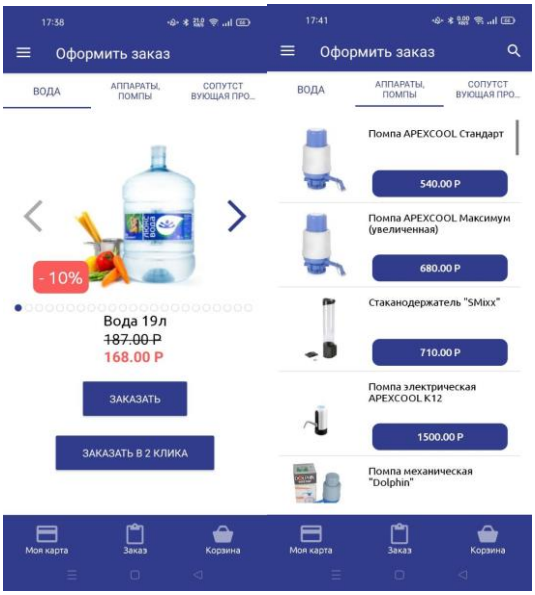


Рисунок 1.3 – Сторінка замовлення

Якщо натиснути на товар, то відкриється вікно з детальною інформацією про даний продукт. В ньому містяться дані про видобуток води, рекомендації по застосуванню, склад та характеристики (рис. 1.4).

Рисунок 1.4 – Сторінка про товар

Перейшовши на сторінку з акціями, клієнт може ознайомитися, за якими умовами він може отримати знижку на покупку. Сторінка з магазинами показує список магазинів та їх місцерозташування на карті. Сторінка з кошиком містить в собі товари, які додав користувач під час покупок (рис. 1.5). Там є можливість змінити кількість товару, чи видалити товар із замовлення.

Рисунок 1.5 – Кошик

Щоб оформити замовлення, потрібно натиснути на кнопку “Замовити”. Після чого відкриється нова сторінка з формою, яка містить наступні поля: П.І.Б., місто, адреса доставки, дім, квартира, коментар до адреси, номер телефону, спосіб оплати та час доставки (рис. 1.6).

Рисунок 1.6 – Замовлення

Дизайн додатку є мінімалістичним, винонаним у біло-синій гаммі. Вся інформація доступна з початкової сторінки. Перехід між сторінками відбувається досить швидко, але не зовсім плавно. Змінити дизайн не є можливим.

Після запуску додатку “ЯП Вода” відкривається сторінка з каталогом (рис. 1.7).

Рисунок 1.7 – Головна сторінка додатку «ЯП Вода»

Всі товари знаходяться в одному списку, але для більш швидкого переходу по категоріям зверху сторінки знаходяться посилання. При натисканні на товар відкривається сторінка з його коротким описом. Щоб додати продукт в кошик, потрібно натиснути відповідну кнопку, на місці якої з’явиться нова кнопка з вибором кількості одиниць, яку купляє замовник.

Через бічне меню користувач може зайти в особистий кабінет, розіслати посилання на компанію друзям, відкрити сторінку “Про компанію”, на якій знаходяться контактні дані та коротка інформація про продавця (рис. 1.8). Окрім цього, доступна сторінка зворотного зв’язку, де можна відправити побажання, скаргу чи відгук, заповнивши форму, а саме поля з ім’ям, e-mail, номером телефону та, власне, самим повідомленням (рис. 1.9).

Рисунок 1.8 – Про продавця

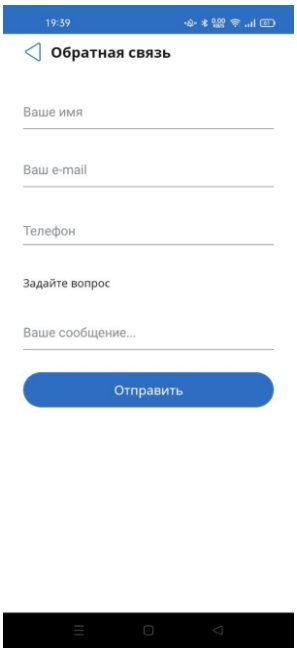


Рисунок 1.9 – Сторінка зворотного додатку

Дизайн виконаний у тому ж самому мінімалістичному стилі, як і у попереднього аналогу. Функціонал додатку розрахований на швидке придбання товару, так як одразу відкривається каталог, а інших розділів не так багато.

Третій додаток-аналог «Big bottle» (рис. 1.10) представляє собою копію додатку «ЯП Вода», за виключенням того, що різноманіття товарів тут більше. Окрім цього, акції виведені одразу на головний екран, щоб користувач одразу дізнавався про можливі знижки. Дизайн, окрім вищеописаних функціональних доповнень, ідентичний.

Рисунок 1.10 – Сторінка додатку «Big bottle»

Після процесу порівняння аналогів була створена порівняльна таблиця (табл. 1.1), де описаний функціонал, який мають додатки. Але тестування додатків було лише з точки зору клієнта, а не працівника компанії, так як немає доступу до схованого корпоративного функціоналу програми (якщо він присутній).

Таблиця 1.1 – Порівняльна таблиця додатків-аналогів

Характеристика	**Люкс Вода**	**ЯП Вода**	**Big Bottle**
Унікальний дизайн	+	-	-
Швидкість роботи	+	+	+
Робота без авторизації	+	-	-
Зручна навігація	-	+	+
Особистий кабінет	+	+	+
Організація доставки	-	-	-

2 ПОСТАНОВКА ЗАДАЧІ ТА МЕТОДИ ДОСЛІДЖЕННЯ

2.1 Постановка задачі

Метою даної роботи є розробка інформаційної системи у вигляді мобільного додатка на операційній системі Android для організації замовлення питної води. Інформаційна система буде реалізована у вигляді мобільного додатка, тому що мобільний пристрій майже завжди знаходиться поруч, тобто, функціонал може використовуватися в будь-який час та в бідь-якому місці. Даний продукт буде використовуватися для більш зручного замовлення питної води та супутньої продукції через прискорення часу оформлення замовлення, яке, окрім цього, не потребує ніяких додаткових дзвінків до постачальника. Також додаток буде корисний і для робітників компанії, що постачає воду, а саме для водіїв-кур'єрів, яким буде будуватися отпимальний маршрут для розвезення товарів, та оператора, який матиме змогу контролювати список замовлень.

Програмою будуть користуватися наступні категорії користувачів:

– клієнт;
– кур'єр підприємства;
– адміністратор підприємства.

Користувач матиме доступ до наступних функцій:

– перегляд списку товарів;
– додавання товарів до кошику;
– оформлення замовлення доставки товарів;
– авторизація та реєстрація у особовий кабінет;
– перегляд попередніх власних замовлень;
– відміна поточних замовлень.

Кур’єр підприємства матиме доступ до подальших функцій:

– перегляд списку актуальних замовлень;

– перегляд замовлень на карті;

– створення оптимального маршруту доставки товарів по точкам замовлень клієнтів;

– зміна статусу замовлення.

Адміністратор підприємства матиме доступ до таких функцій:

– перегляд списку всіх замовлень;

– додавання та редагування товарів;

– додавання чи видалення кур’єру;

– зміна статусу замовлення.

Мобільний додаток буде написано мовою Java для операційної системи Android. Працювати додаток буде на версіях ОС вище 4.0.3, яку має більшість з усіх пристроїв. Інтерфейс можна буде охарактеризувати як сучасний, не перезавантажений та швидкий.

2.2 Технології та засоби реалізації

Середовищем для розробки мобільного додатку було обрано Android Studio, яке створено для розробки під операційну систему Android. Як офіційне інтегроване середовище розробки (IDE) операційної системи Android, Android Studio добре обладнана для швидкої розробки, забезпечуючи високоякісний вихід додатків на всіх пристроях Android [23].

Основні переваги Android Studio:

– Швидке кодування та швидка ітерація.

Завдяки платформі IntelliJ IDEA ця IDE забезпечує швидке автодоповнення коду та миттєву оцінку робочого процесу. Також існують

певні функції Android Studio, такі як підтримка коду для змін і редактор коду для оптимізації виведення коду.

Android Studio дозволяє розробникам швидко вносити зміни без повного перезапуску програми. Це забезпечує гнучкість для внесення невеликих змін до програми, поки програма все ще працює.

Інтуїтивно зрозумілий редактор коду Android Studio є ключовою функцією, яка забезпечує одну з основних переваг Android Studio, як більш швидке програмування. У той же час він забезпечує найсучасніший рефакторинг, завершення коду та аналіз коду.

– Швидкий і багатофункціональний емулятор.

Android Studio постачається з емулятором, який допомагає запускати програму швидше, ніж фактичний пристрій. Емулятор, дозволяючи тестувати додаток на кількох пристроях, включаючи телефони, планшети, Android Wear і Android TV, може моделювати кілька різних апаратних функцій, як-от GPS, кілька сенсорних вводів, датчики руху та прискорення тощо.

– Надійні механізми тестування.

Android Studio пропонує багато інструментів і фреймворків для тестування, які допомагають тестувати програми Android за допомогою функціональних інструментів тестування інтерфейсу користувача. Переваги Android Studio також показані з усіма видами розширених інструментів і фреймворків для будь-яких цілей. Тести, які проводяться цими інструментами, можуть проводитися на реальних пристроях, емуляторах або надійних інтеграційних середовищах, а також за допомогою тестової лабораторії Firebase.

– Підтримка Firebase та інтегрована хмара.

Android Studio постачається з Firebase Assistant, який дозволяє підключати будь-яку програму до сервера Firebase, окрім додавання багатьох основних послуг, таких як аналітика додатків, аутентифікація, повідомлення сповіщень та деякі інші. Android Studio також допомагає інтегрувати програму з платформою Google Cloud.

– Редактор макетів.

Android Studio надає інструмент візуального редактора перетягування для роботи з файлами XML. Це допомагає легко створити абсолютно новий макет програми. Редактор макетів Android Studio, створений синхронно з ConstraintLayout API, дозволяє створювати макет, налаштований для різних розмірів екрана. Це забезпечує підтримку оптимізованих підходів до проектування на основі вимог пристрою.

– Потужна система розвитку.

До переваг Android Studio можна віднести кілька розширених функцій, які допомагають автоматизувати процеси, керувати залежностями та налаштовувати конфігурацію раз і назавжди. Як локальні, так і розміщені бібліотеки можуть бути включені в проект, і існує величезна кількість налаштувань для проектів програми.

Для збереження данних додатку було обрано сервіс Google Firebase саме через зручність його інтеграції під час розробки.

Firebase — це сервіс, корисний для створення портативних і веб-додатків, яким потрібен режим реального часу бази даних, що означає, що коли один користувач оновлює запис у базі даних, оновлення має бути передано кожному окремому користувачу миттєво [24]. Він дає базову та уніфіковану платформу для багатьох програм, а також ряд інших функцій Google, які входять до складу сервіс. Firebase виконує більшість роботи на стороні сервера, коли справа доходить до розробки додатків. Існує численні елементи, які роблять Firebase таким важливим інструментом у розробці з точки зору розробника.

Найважливішими компонентами або послугами, які пропонує Firebase, та будуть використані під час розробки інформаційної системи, є:

- База даних реального часу — це хмарна база даних. Дані зберігаються у форматі JSON і синхронізуються постійно кожному асоційованому клієнту. Коли будь-яка кросплатформна програма розроблена за допомогою iOS, Android і JavaScript SDK, більша частина попиту користувача базується на

одному екземплярі бази даних у реальному часі, і цей екземпляр оновлюється за допомогою кожних нових даних. Ця функція дозволяє розробникам пропускати етап розробки бази даних, а Firebase обробляє більшість бекенд для програм. Це дає адаптивну мову правил, засновану на виразах, щоб визначити, як дані мають бути організовані та коли інформацію можна переглядати або складати.

– Сховище. Firebase Storage було розроблено для розробників додатків, яким потрібно зберігати та обслуговувати створений користувачами вміст, наприклад, фотографії або будь-який інший файл. Він забезпечує безпечну передачу та завантаження документів для програм Firebase, незалежно від якісті мережі. Firebase Storage підтримується Google Cloud Storage, потужною, базовою та економічно ефективною службою зберігання об’єктів.

Для роботи із зображеннями буде використана бібліотека для завантаження та кешування зображень для Android під назвою Picasso. Дана бібліотека вирішує наступні недоліки роботи з зображеннями на Android:

– скасування завантаження в адаптері;
– перетворення зображень з мінімальним використанням пам'яті;
– автоматичне кешування пам'яті та диска.

3 ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ ОРГАНІЗАЦІЇ ЗАМОВЛЕННЯ БУТИЛЬОВАНОЇ ПИТНОЇ ВОДИ

3.1 Структурно-функціональне моделювання організації замовлення бутильованої питної води

Основним методом для моделювання структури та функціоналу процесів на основі структурного та об’єктно-орієнтованого підходу є функціональне моделювання SADT (IDEF0).

Метою даного структурно-функціонального моделювання є декомпозиція основного процесу на менші частини за певними процедурами та правилами [25].

IDEF0 – модель [26] демонструє специфікацію та взаємозв’язок усіх дій, які відбуваються у процесі між собою. Контекстну діаграму головного процесу організації замовлення бутильованої питної води показано на рисунку 3.1.

Об’єкти, що впливають на процес, позначаються стрілками:

− “Вхід” (ліва сторона): запит на замовлення товарів;

− “Управління” (верхня сторона): інформація про товари, карта міста, вимоги до мобільного додатку;

− “Механізми” (нижня сторона): користувач, працівник, інформаційна система, база даних;

− “Вихід” (права сторона): виконане замовлення, побудований маршрут доставки.

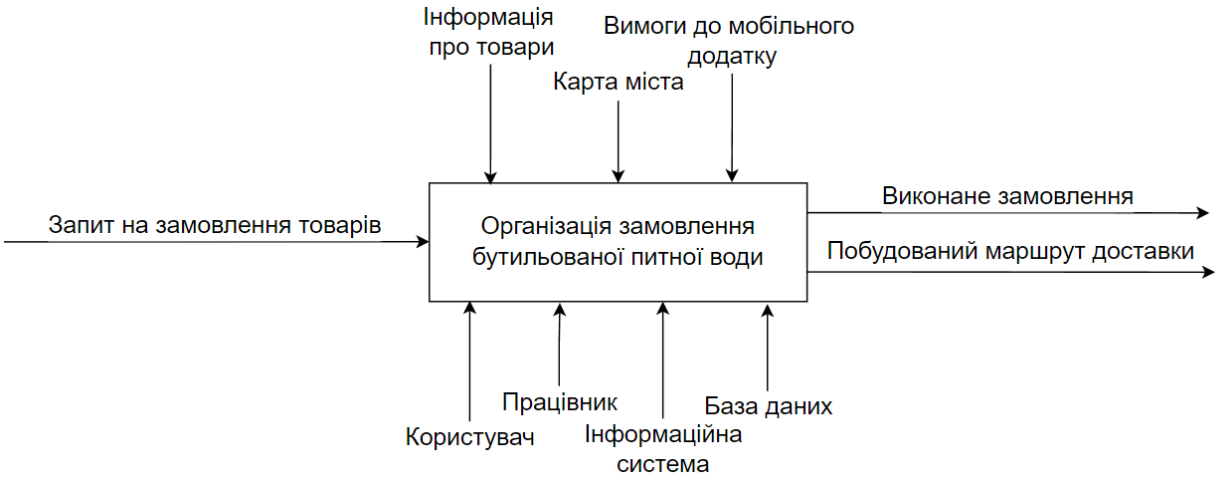


Рисунок 3.1 – Контекстна діаграма організації замовлення бутильованої питної води в нотації IDEF0

В контекстній діаграмі декомпозиція виконується після того, як була описана головна функція [27]. На цьому етапі потрібно визначити функції, які можуть впливати на основну.

У декомпозиції моделі головної функції інформаційної системи було визначено п’ять блоків:

– оформлення замовлення;

– підтвердження замовлення;

– побудова маршруту доставки;

– виконання замовлення;

– зміна статусу замовлення на виконане.

Дуги та зв’язки з батьківської діаграми переходять і на діаграму декомпозиції, але окрім існуючих додаються і нові для деталізації функціонування інформаційної системи (рис. 3.2).

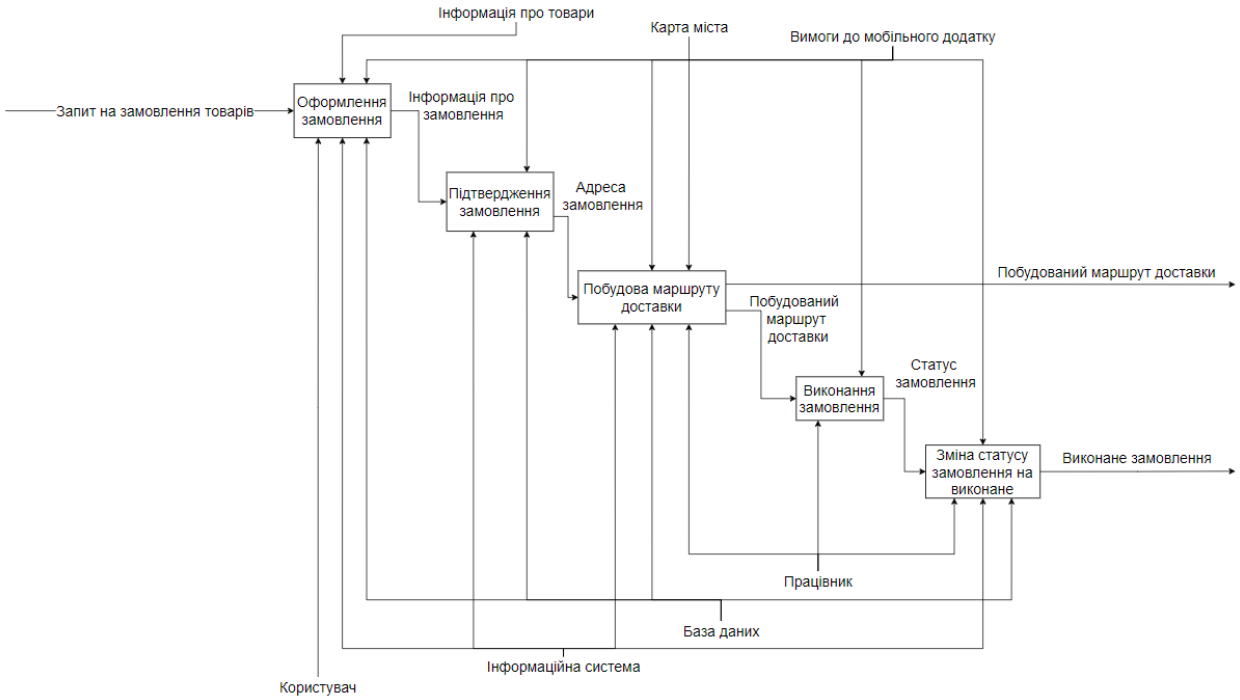


Рисунок 3.2 – Діаграма декомпозиції

Блок “Оформлення замовлення”:

− Вхідною стрілкою є “Запит на замовлення товарів”;

− Вихідною стрілкою – “Інформація про замовлення”;

− Стрілки контролю – “Інформація про товари”, “Вимоги до мобільного додатку”;

− Стрілки механізму – “Користувач”, “Інформаційна система ”, “База даних”.

Блок “Підтвердження замовлення”:

− Вхідною стрілкою є “Інформація про замовлення”;

− Вихідною стрілкою – “Адреса замовлення”;

− Стрілка контролю – “ Вимоги до мобільного додатку”;

− Стрілки механізму –“Інформаційна система ”, “База даних”.

Блок “Побудова маршруту доставки”:

− Вхідною стрілкою є “Адреса замовлення”;

− Вихідною стрілкою – “Побудований маршрут доставки”;

− Стрілки контролю – “Карта міста”, “Вимоги до мобільного додатку”;

− Стрілки механізму – “Працівник”, “Інформаційна система ”, “База даних”.

Блок “Виконання замовлення”:

− Вхідною стрілкою є “Побудований маршрут доставки”;

− Вихідною стрілкою – “Статус замовлення”;

− Стрілка контролю – “Вимоги до мобільного додатку”;

− Стрілка механізму – “Працівник”.

Блок “Зміна статусу замовлення на виконане”:

− Вхідною стрілкою є “Статус замовлення”;

− Вихідною стрілкою – “Виконане замовлення”;

− Стрілка контролю – “Вимоги до мобільного додатку”;

− Стрілки механізму – “Працівник”, “Інформаційна система ”, “База даних”.

3.2 Моделювання варіантів використання інформаційної системи організації замовлення бутильованої питної води

Діаграми варіантів використання описує функціональне призначення системи. Тобто, що система під час своєї роботи буде робити.

Суть даної діаграми полягає в представленні проектованої системи як сукупність сутностей або суб'єктів, які взаємодіють із системою через варіанти використання [28].

У даному випадку актором називають будь-яку сутність, що взаємодіє із системою зовні. А для опису послуг, що актору надає система використовують варіанти використання. Система виконує певний набір дій під час взаємодії з актором – саме ці дії і визначає кожен варіант використання [29].

Список акторів наведено в табл. 3.1. Варіанти використання та їх опис розглянуто в табл. 3.2. Взаємодію акторів з інформаційною системою зображено на рис. 3.3

Таблиця 3.1 – Опис акторів

№	Актор	Опис діяльності
1	Користувач	Користувач, що має можливість створювати та редагувати власні замовлення в інформаційній системі.
2	Кур’єр	Користувач, що має можливість переглядати замовлення та змінювати їх статус в інформаційній системі.
3	Адміністратор	Користувач, що має можливість додавати, переглядати усі замовлення та змінювати їх статус.

Таблиця 3.2 – Опис варіантів використання

№	Назва варіанту використання	Опис
1	Авторизація	Дозволяє усім користувачам авторизуватися на сайті.
2	Реєстрація	Дозволяє простим користувачам реєструватися на сайті.
3	Замовлення товарів	Дозволяє користувачам створювати нові замовлення товарів.
4	Редагування замовлення	Дозволяє користувачам редагувати власні замовлення, а для кур’єра та адміністратора всі існуючі замовлення товарів.
5	Побудова маршруту доставки	Будує маршрут доставки на карті по адресам замовлень.
6	Зміна статусу замовлення	Дозволяє кур’єру та адміністратору редагувати статус замовлення товарів.

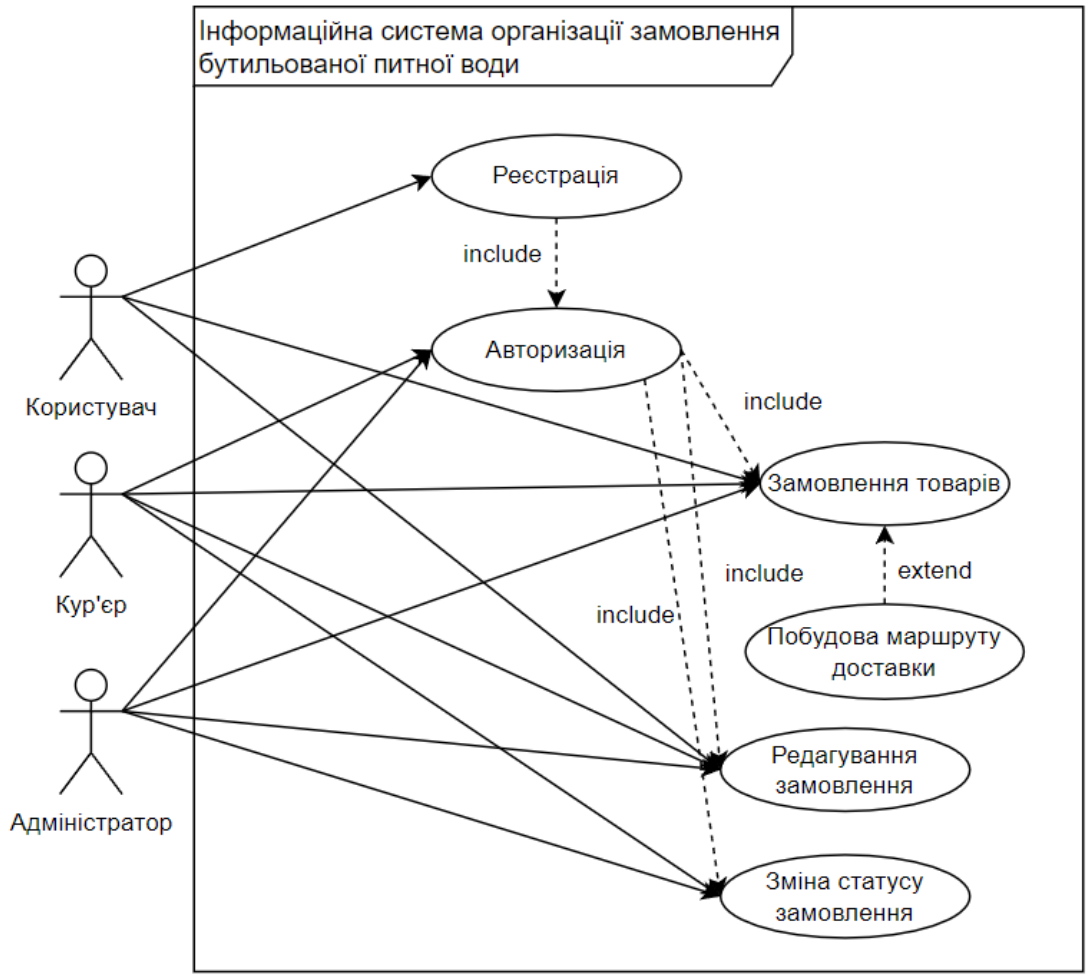


Рисунок 3.3 – Діаграма варіантів використання інформаційної системи організації замовлення бутильованої питної води

3.3 Моделювання діаграм діяльності

Діаграми діяльності використовуються для опису процедур чи логіки про бізнес-процеси. Окрім цього, вони візуалізують потік управління процесом [30]. Можна розглядати випадок, коли один процес складається з більш ніж одного варіанту використання. У цьому випадку, один вузол також може представляти лише один варіант використання, який втілює кроки нижнього рівня в загальній діяльності відповідно до специфікації UML. Все

це лише підтверджує більший або рівний вплив діаграми діяльності на модель у порівнянні з впливом варіантів використання.

На рис. 3.4-3.8 зображено діаграми діяльності модулів інформаційної системи організації замовлення бутильованої питної води.

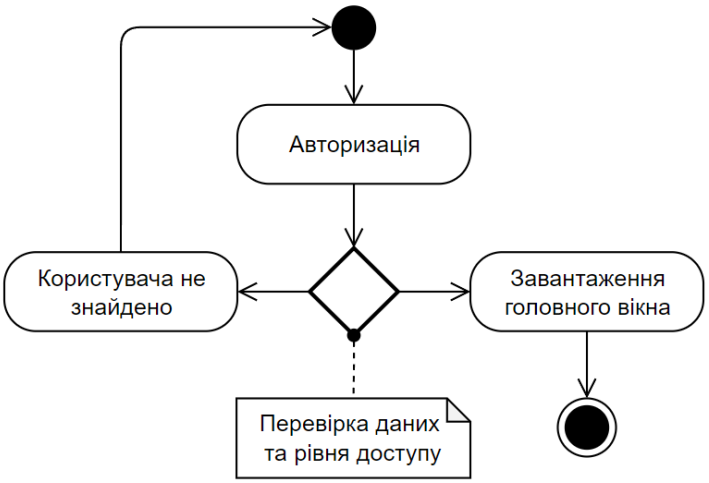


Рисунок 3.4 – Діаграма діяльності модулю авторизації

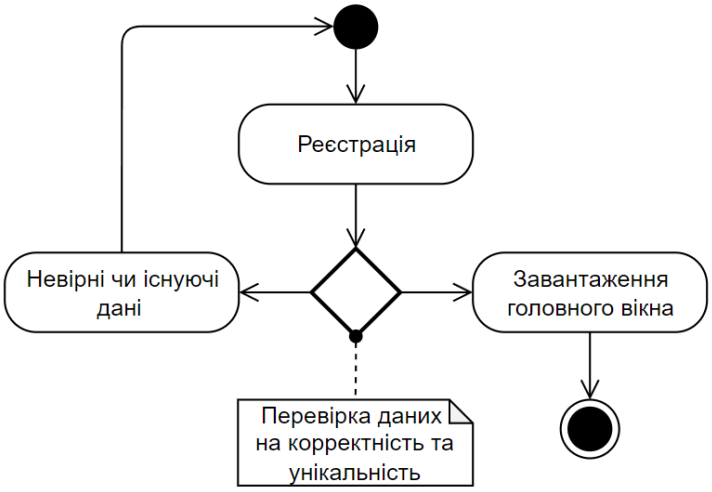


Рисунок 3.5 – Діаграма діяльності модулю реєстрації

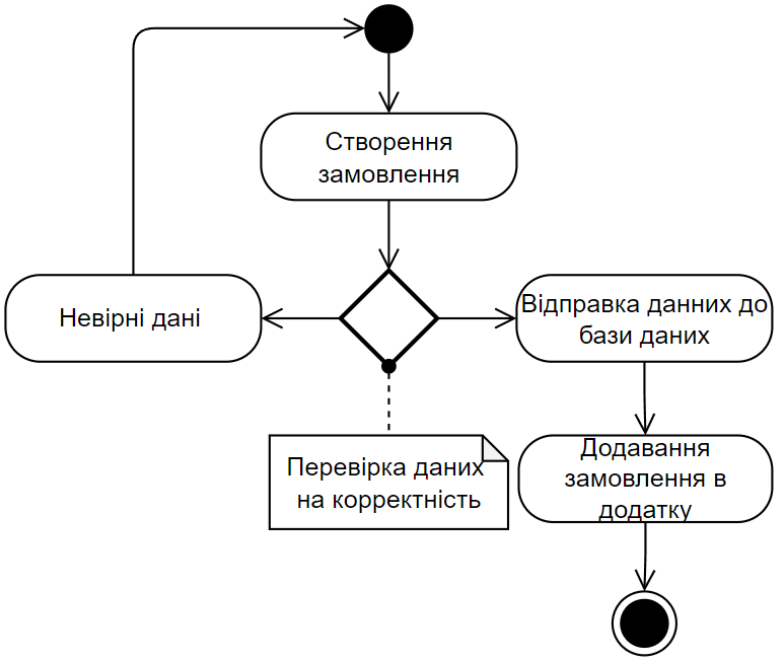


Рисунок 3.6 – Діаграма діяльності модулю створення замовлення

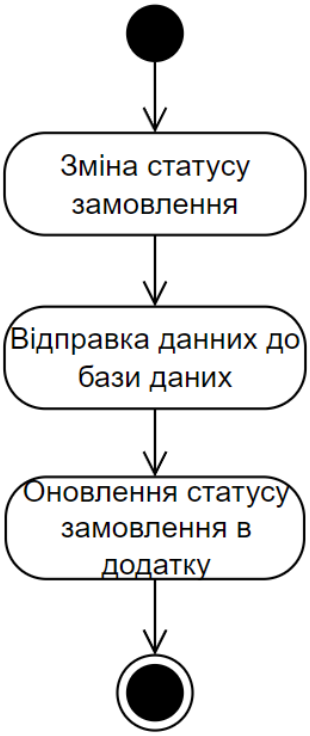


Рисунок 3.7 – Діаграма діяльності модулю зміни статусу замовлення

3.4 Проектування бази даних

База даних являє собою сховище для певного набору файлів даних, що занесені в комп'ютер. Отже, в інформаційній системі база даних відповідальна за процес збереження інформації. Система при цьому здійснює пошук данної інформації в цій базі даних. Метою проектування бази даних є вірне відображення предметної області [31].

Збереження даних буде в форматі JSON-файлу. JSON – простий спосіб зберігати та передавати структуровані дані, оснований на використанні тексту. За допомогою простого синтаксису можна легко зберігати дані, від одного числа до рядків, масивів та об'єктів у простому тексті [32].

Рядок JSON повинен містити в собі об'єкт, тобто, асоціативний масив пар ім'я/значення.

У процесі проектування бази даних було виділено наступні об’єкти:

– Товари (Products);
– Користувачі (Users);
– Кошик (Cart List);
– Замовлення (Orders);
– Архів (Archive);

На рис. 3.8-3.12 приведено об’єкти бази даних. У таблиці 3.3 представлено детальний опис полів об’єктів.

```
"Archive" : {
  "Order_ID" : {
    "Products" : {
      "Product_ID" : {
        "date" : "Дата створення товару",
        "goods_id" : "Ідентифікатор товару",
        "goods_name" : "Назва товару",
        "goods_price" : "Ціна товару",
        "quantity" : "Кількість товару",
        "time" : "Час створення товару"
      }
    },
    "address" : "Адреса доставки замовлення",
    "comment" : "Коментар до замовлення",
    "date" : "Дата доставки",
    "name" : "Ім'я замовника",
    "orderid" : "Ідентифікатор замовлення",
    "phone" : "Номер телефону замовлення",
    "phoneID" : "Ідентифікатор користувача",
    "state" : "Статус замовлення",
    "time" : "Години доставки",
    "totalprice" : "Загальна ціна замовленя"
  }
```

Рисунок 3.8 – Об’єкт Архів

```
"Cart List" : {
  "User View" : {
    "User_id" : {
      "Products" : {
        "Product_ID" : {
          "date" : "Дата створення товару",
          "goods_id" : "Ідентифікатор товару",
          "goods_name" : "Назва товару",
          "goods_price" : "Ціна товару",
          "quantity" : "Кількість товару",
          "time" : "Час створення товару"
        }
      }
    }
  }
```

Рисунок 3.9 – Об’єкт Кошик

```
"Orders" : {
  "Order_ID" : {
    "Products" : {
      "Product_ID" : {
        "date" : "Дата створення товару",
        "goods_id" : "Ідентифікатор товару",
        "goods_name" : "Назва товару",
        "goods_price" : "Ціна товару",
        "quantity" : "Кількість товару",
        "time" : "Час створення товару"
      }
    },
    "address" : "Адреса доставки замовлення",
    "comment" : "Коментар до замовлення",
    "date" : "Дата доставки",
    "name" : "Ім'я замовника",
    "orderid" : "Ідентифікатор замовлення",
    "phone" : "Номер телефону замовлення",
    "phoneID" : "Ідентифікатор користувача",
    "state" : "Статус замовлення",
    "time" : "Години доставки",
    "totalprice" : "Загальна ціна замовленя"
  }
```

Рисунок 3.10 – Об’єкт замовлень

```
"Products" : {
  "Product_ID" : {
    "date" : "Дата створення товару",
    "goods_description" : "Опис товару",
    "goods_id" : "Ідентифікатор товару",
    "goods_image" : "Посилання на зображення товару",
    "goods_name" : "Назва товару",
    "goods_price" : "Ціна товару",
    "time" : "Час створення товару"
  }
```

Рисунок 3.11 – Об’єкт товарів

```
"Users" : {
  "User_id" : {
    "address" : "Основна адреса користувача",
    "address2" : "Додаткова адреса користувача",
    "level" : "Рівень доступу користувача",
    "name" : "Ім'я користувача",
    "password" : "Пароль для входу",
    "phone" : "Номер телефону користувача",
    "phoneOrder" : "Номер телефону для замовлень"
  }
}
```

Рисунок 3.12 – Об’єкт користувачів

Таблиця 3.3 – Опис полів об’єктів бази даних

Об’єкт	Назва поля	Зміст	Тип	Значення за замовчуванням
Archive та Orders	address	Адреса доставки замовлення	text(50)	Відсутнє
	comment	Коментар до замовлення	text(40)	Відсутнє
	date	Дата доставки	date	Відсутнє
	name	Ім'я замовника	text(40)	Відсутнє
	orderid	Ідентифікатор замовлення	datetime	Відсутнє
	phone	Номер телефону замовлення	int	Відсутнє
	phoneID	Ідентифікатор користувача	int	Відсутнє
	state	Статус замовлення	text(30)	“Нове замовлення”
	time	Години доставки	text(15)	Відсутнє
	totalprice	Загальна ціна замовленя	int	Відсутнє
Products та Cart List	date	Дата створення	date	Відсутнє
	goods_description	Опис товару	text(100)	Відсутнє
	goods_id	Ідентифікатор товару	datetime	Відсутнє
	goods_image	Посилання на зображення товару	text	Відсутнє
	goods_name	Назва товару	text(50)	Відсутнє
	goods_price	Ціна товару	int	Відсутнє
	time	Час створення	time	Відсутнє

Продовження таблиці 3.3 – Опис значень полів бази даних

Об’єкт	Назва поля	Зміст	Тип	Значення за замовчуванням
Users	address	Основна адреса користувача	text(50)	Відсутнє
	address2	Додаткова адреса користувача	text(50)	Відсутнє
	level	Рівень доступу користувача	int	“0”
	name	Ім'я користувача	text(40)	Відсутнє
	password	Пароль для входу	text(20)	Відсутнє
	phone	Номер телефону користувача	int	Відсутнє
	phoneOrder	Номер телефону для замовлень	int	Відсутнє

4 РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ ОРГАНІЗАЦІЇ ЗАМОВЛЕННЯ БУТИЛЬОВАНОЇ ПИТНОЇ ВОДИ

4.1 Архітектура інформаційної системи

Перед початком розробки потрібно визначити архітектуру.

Архітектура інформаційної системи (рис. 4.1) була побудована на основі шаблону MVVM (Model, View, View Model). Архітектура MVVM, як і зазначено в її назві, складається з трьох компонентів [33]. Компонент View показує інтерфейс користувача програми. Model представляє дані. View-Model призначений для керування станом перегляду. Він передасть дані та операції для перегляду, а також керує логікою та поведінкою представлення [34].

Схема архітектури інформаційної системи зображена на рис. 4.1.

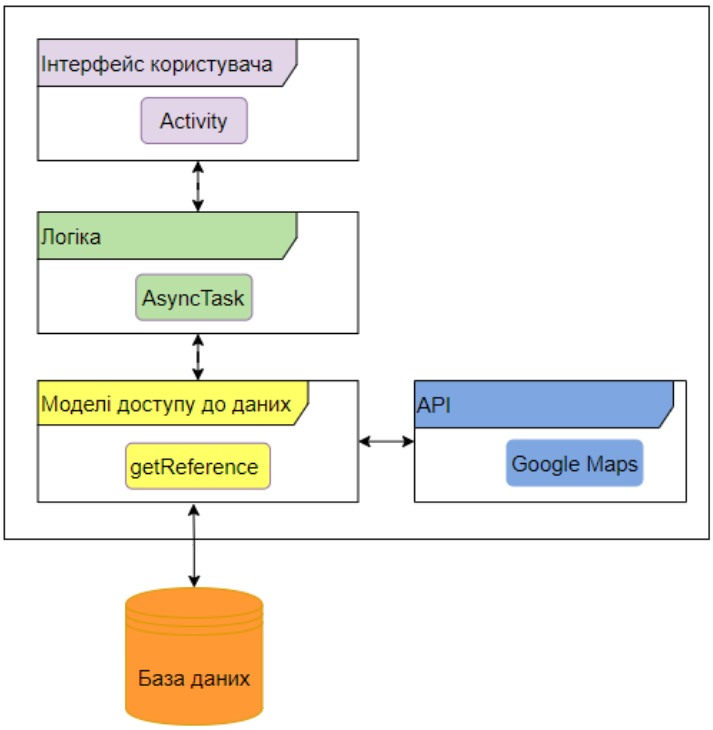


Рисунок 4.1 – Архітектура розроблюваної інформаційної системи

4.2 Розробка інформаційної системи

База даних була заповнена згідно з результатами проектування. На рис. 4.2 показана заповнена база даних Firebase, дані у яку заносилися під час роботи мобільного додатку.

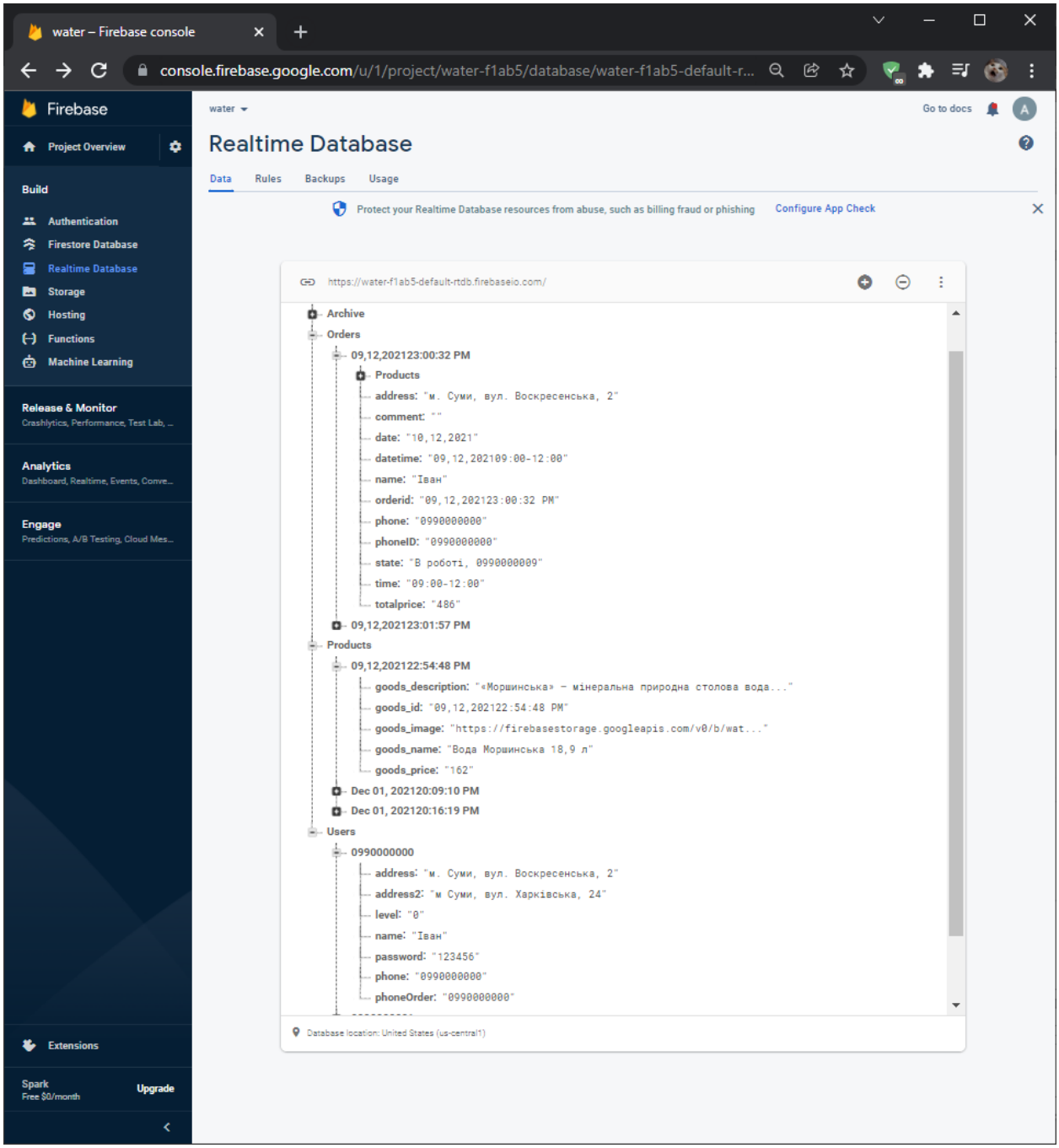


Рисунок 4.2 – База даних

Опис класів та методів, які створювалися під час розробки мобільного додатку наведено у табл. 4.1.

Таблиця 4.1 – Опис класів та методів

Назва класу	Призначення класу
HomeActivity	Призначений для відображення списку товарів у каталогу.
AdminActivity	Призначений для групування можливостей адміністратора.
AdminAddNew ProductActivity	Призначений для додавання нового товару адміністратором.
AdminManageActivity	Призначений для редагування чи видалення існуючого товару адміністратором.
AdminManage StuffActivity	Призначений для додавання чи видалення кур’єрів адміністратором.
AdminOrdersActivity	Призначений для перегляду списку замовлень адміністратором.
AuthActivity	Призначений для авторизації усіх користувачів, після якої вони потрапляють на наступний екран згідно зі свого рівня доступу.
CartActivity	Призначений для відображення списку товарів у кошику користувача.
ConfirmOrderActivity	Призначений для вводу даних до замовлення та підтвердження замовлення користувачем.
CourierActivity	Призначений для відображення актуального списку замовлень на поточний день.
MapsActivity	Призначений для відображення адреси замовлень на карті.
ProductDetailsActivity	Призначений для відображення детальної інформації про товар.

Продовження таблиці 4.1 – Опис класів та методів

Назва класу	Призначення класу
RegActivity	Призначений для реєстрації нових користувачів.
SettingsActivity	Призначений для зміни контактних даних.
SplashScreen	Призначений для запуску заставки під час відкриття додатку.
UserOrder ProductsActivity	Призначений для відображення списку товарів у замовленні.
UserOrdersActivity	Призначений для відображення списку замовлень користувача.

Основні методи та їх опис, які були використані у різних класах, представлено в табл. 4.2.

Таблиця 4.2 – Основні методи класів

Клас	Метод	Призначення
HomeActivity	onCreate()	Встановлює файл розмітки, визначає можливості адміністратора та коритсувача.
	onCreateOptionsMenu()	Додає в меню іконку пошуку.
	process_search()	Виконує пошук по назві товару в базі данних та оновлює список.
	onStart()	Виводить список товарів з бази даних.
	onNavigationItemSelected()	Виконує дії по натисканню на кнопки з бокового меню.

Продовження таблиці 4.2 – Основні методи класів

Клас	Метод	Призначення
ProductDetails Activity	addingToCartList()	Додає товар та вибрану кількість у базу даних.
	getProductDetails()	Виводить всю інформацію про товар з бази даних по ідентифікатору.
AdminAddNew ProductActivity	OpenGallery()	Відкриває галерею мобільного телефону для вибору зображення.
	ValidateProductData()	Перевіряє валідність введених даних про товар.
	SaveProductInfoToDatabase()	Заносить дані про товар в базу даних.
AdminOrders Activity	moveFireBaseRecord()	Переносить запис бази даних з однієї сутності в іншу.
CartActivity	getTotalValue	Рахує загальну вартість усіх товарів.
ConfirmOrder Activity	confirmOrder()	Створює ідентифікатор замовлення, заносить інформацію про замовлення в базу та викликає метод для переміщення запису в БД.
RegActivity	CreateAccount()	Створює новий аккаунт.
SettingsActivity	updateOnlyUserInfo()	Оновлює дані користувача.
SplashScreen	AllowAccess()	Перевіряє наявність сессії та направляє користувача на певний екран згідно рівня доступу.

4.3 Використання інформаційної системи замовлення питної бутильованої води

Після запуску програми відображається екран з анімацією (рис. 4.3). Анімація триває 2 секунди.

Рисунок 4.3 – Екран завантаження

Після цього для нового користувача відкривається головний екран, на якому знаходиться список товарів (рис. 4.4). Даний список відображає фотографію товару, його назву та ціну. Щоб подивитися на інші товари, користувач потрібен прогорнути список вниз.

Рисунок 4.4 – Каталог товарів

Щоб виконати пошук товарів за назвою, потрібно натиснути на відповідний значок, що знаходиться у правому верхньому куті (рис. 4.5).

Рисунок 4.5 – Значок пошуку

Після натискання на даний значок відкриється текстове поле, в яке і потрібно вводити назву необхідного товару. Результати пошуку будуть автоматично оновлюватися (рис. 4.6).

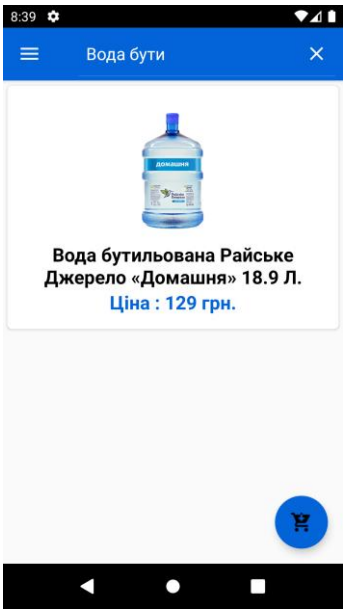


Рисунок 4.6 – Результати пошуку

Натиснувши на товар, відкривається окремий екран з більш детальною інформацією про товар (рис. 4.7). Окрім цього, товар можна додати до кошика, але неавторизованому користувачу дана функція не доступна.

Рисунок 4.7 – Інформація про товар

Також, на головному екрані можна відкрити бічне меню (рис. 4.8), натиснувши на кнопку в лівому верхньому куті чи якщо провести з лівої границі екрану направо. В цьому меню можна здійснювати переходи з одного місця в інше.

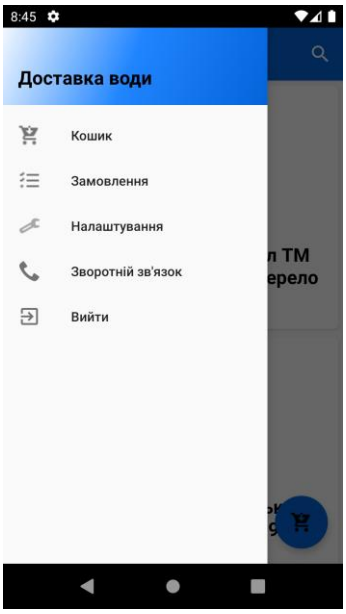


Рисунок 4.8 – Бокове меню

Неавторизований користувач має доступ лише до перегляду каталогу, детальної інформації про товар та зворотнього зв’язку (пункт в боковому меню, після натискання на який відкриється телефон з набраним номером для звернень до компанії-постачальника питної бутильованої води) (рис. 4.9).

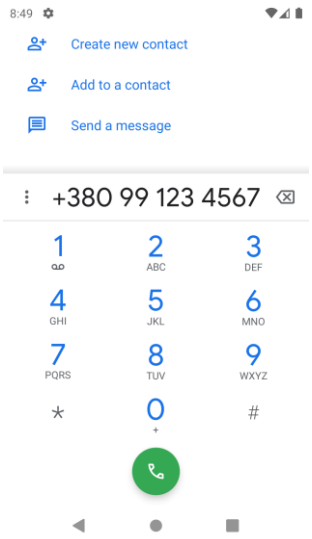


Рисунок 4.9 – Набраний номер

Для переходу на вікно авторизації потрібно натиснути на будь-який кнопку, для якої потрібен вже існуючий аккаунт (перегляд кошику, замовлень, додавання товарів у кошик та інше). Після чого відкриється вікно з авторизацією (рис. 4.10).

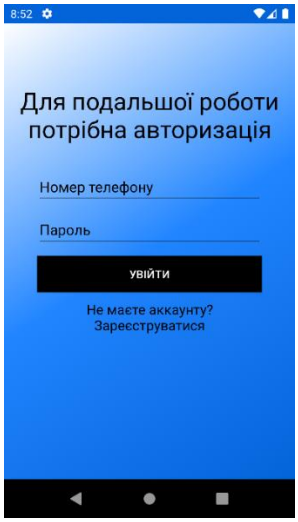


Рисунок 4.10 – Авторизація

Якщо користувач не має аккаунту, то йому потрібно зареєструватися, натиснувши на відповідний надпис знизу форми (рис. 4.11).

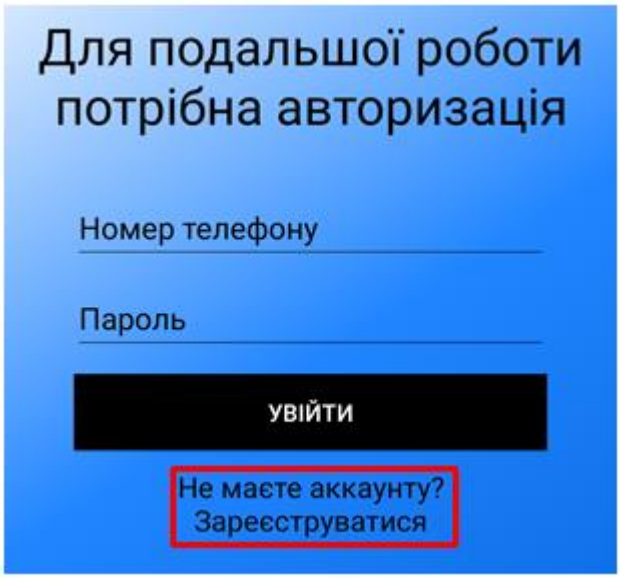


Рисунок 4.11 – Кнопка для реєстрації

Якщо користувач натиснув на даний надпис, то його перенаправить на екран з реєстрацією. Для успішної реєстрації користувачу потрібно ввести вірні дані в форму. Якщо поле залишити порожнім, то користувач отримає відповідне повідомлення (рис. 4.12).

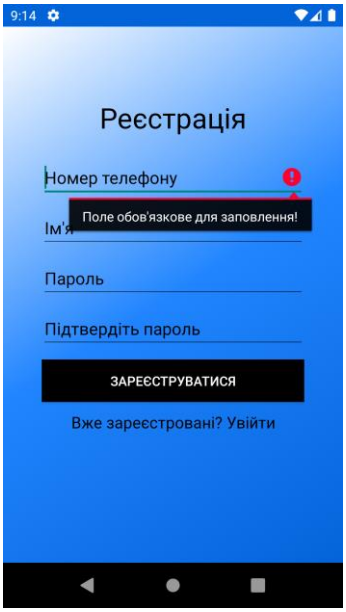


Рисунок 4.12 – Повідомлення про помилку

Також, повідомлення про некоректні дані з’явиться якщо пароль замалий чи не співпадає з паролем для підтвердження (рис. 4.13).

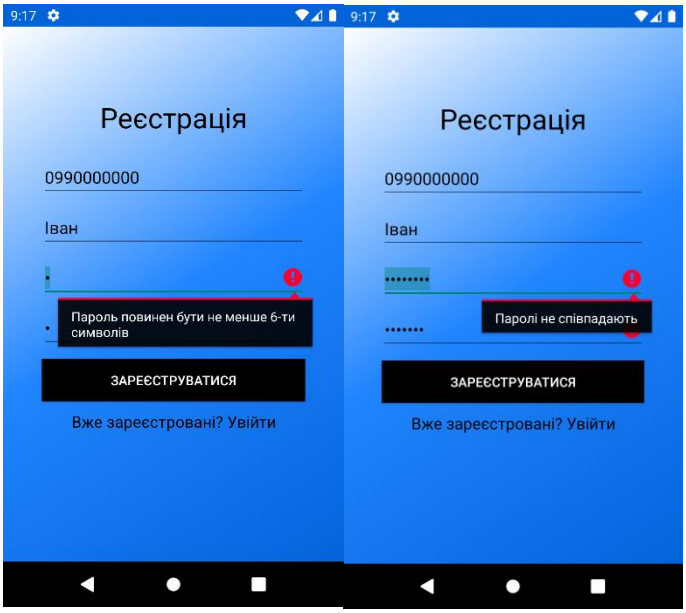


Рисунок 4.13 – Помилка паролів

Після успішної реєстрації користувач отримає повідомлення про це та зможе пройти авторизацію (рис. 4.14).

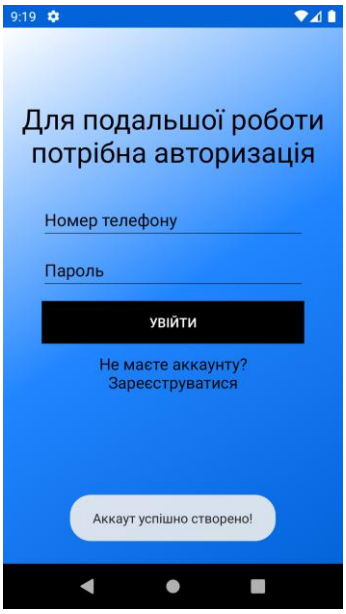


Рисунок 4.14 – Успішна реєстрація

Якщо ввести невірний номер телефону чи пароль під час авторизації, то будуть відповідні повідомлення (рис. 4.15).

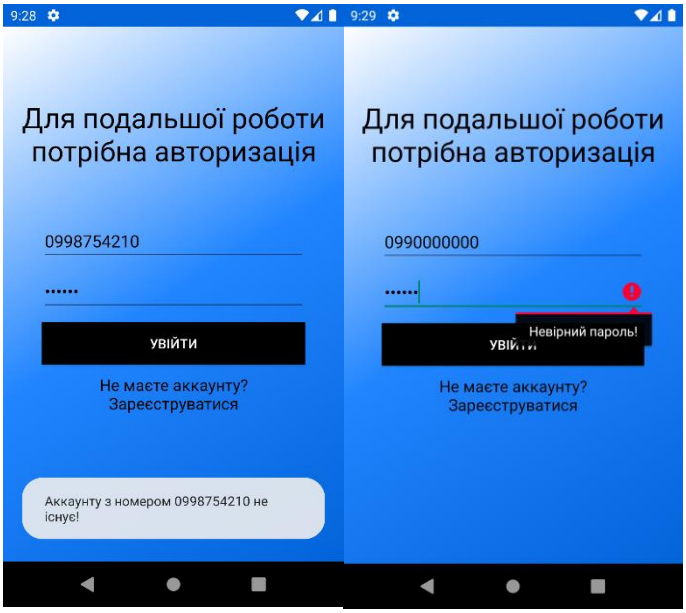


Рисунок 4.15 – Повідомлення про помилку

Після того, як користувач введе вірний номер телефону та пароль, які використовував під час реєстрації, то знову відкриється екран з каталогом. В боковому меню буде відображатися ім’я, яке користувач ввів під час реєстрації (рис. 4.16).

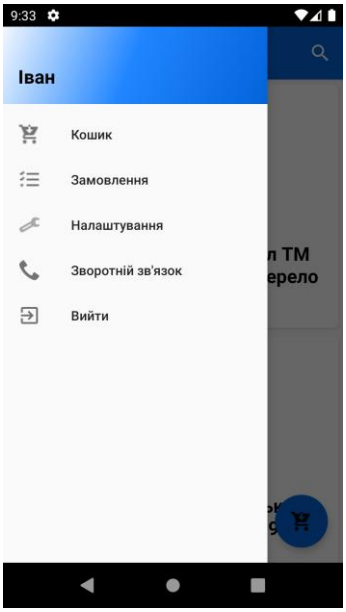


Рисунок 4.16 – Відображення імені

Щоб змінити дані користувача, потрібно натиснути на пункт меню “Налаштування”. Після чого відкриється нове вікно з формою, у яку автоматично завантажуться існуючі дані про користувача (рис. 4.17). Для їх зміни потрібно лише ввести необхідну інфомацію, і натиснувши відповідну кнопку дані оновляться в базі даних.

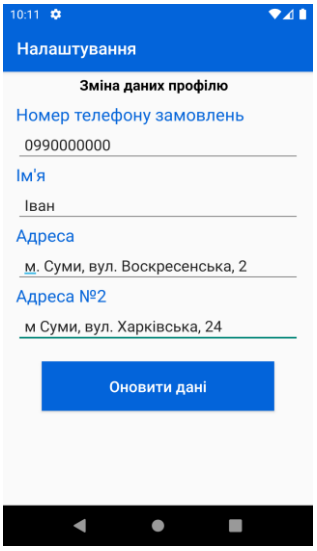


Рисунок 4.17 – Екран налашувань

Щоб додати певний товар до кошика, потрібно на головному екрані-каталозі обрати необхідний продукт, натиснути на нього, та за допомогою кнопки перемикача обрати необхідну кількість (натискаючи на плюс кількість додається, а на мінус-віднімається) (рис. 4.18), а потім натиснути на кнопку “Додати до кошика”.

Рисунок 4.18 – Вибір кількості

Виконавши це, користувач повернеться до каталогу, йому відобразиться повідомлення про успішну операцію (рис. 4.19).

Рисунок 4.19 – Успішна операція

Для того, щоб переглянути вміст кошику, потрібно або натиснути на пункт в меню, або на відповідну кнопку на екрані каталогу (рис. 4.20), що знаходиться в правому нижньому куті.

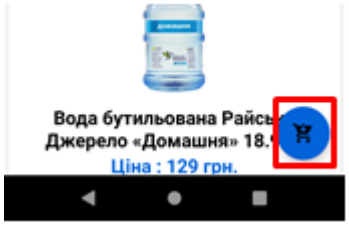


Рисунок 4.20 – Кнопка кошику

У кошику відображається список, а саме назва товару, його кількість та ціна, що є результатом множення кількості одиниць товару на його ціну за одну штуку (рис. 4.21). А знизу екрану порахована загальна ціна усіх товарів у кошику.

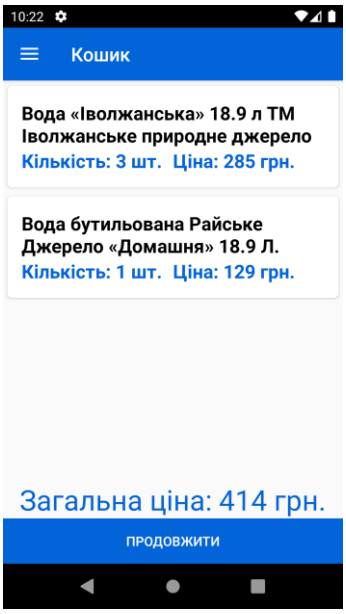


Рисунок 4.21 – Кошик

Натиснувши на товар у кошику, користувач має змогу змінити кількість або видалити товар з кошику (рис. 4.22).

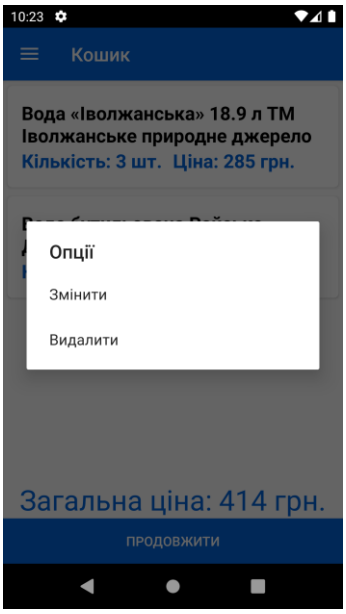


Рисунок 4.22 – Редагування даних у кошику

Але якщо кошик порожній, то буде відображатися відповідне повідомлення з кнопкою, яка перенаправить користувача на каталог (рис. 4.23).

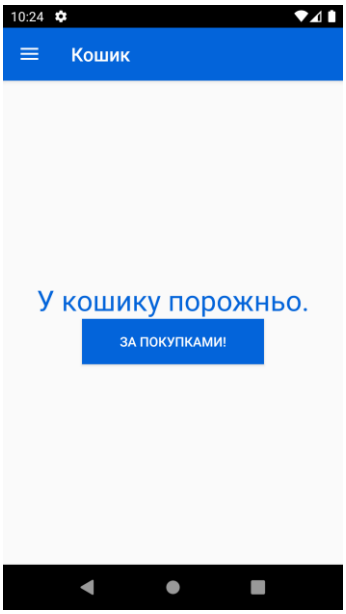


Рисунок 4.23 – Порожній кошик

Щоб підтвердити замовлення, потрібно підтвердити дані для доставки замовлення, які автоматично завантажуються з бази даних. Користувач має

змогу їх змінити за бажанням. Для відправки замовлення, потрібно натиснути на кнопку “Підтвердити замовлення” (рис. 4.24).

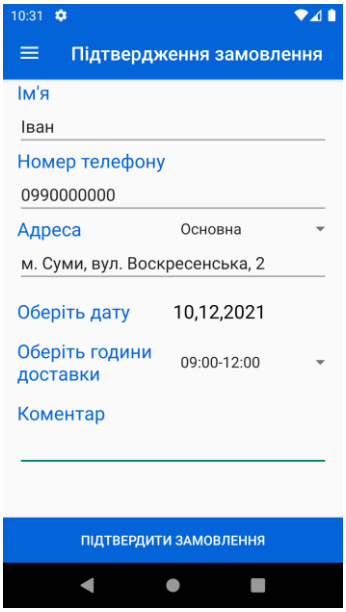


Рисунок 4.24 – Підтвердження замовлення

Щоб переглянути створені замовлення, потрібно натиснути на пункт меню “Замовлення” (рис. 4.25). Там відображена інформація про поточні та архівні замовлення користувача.

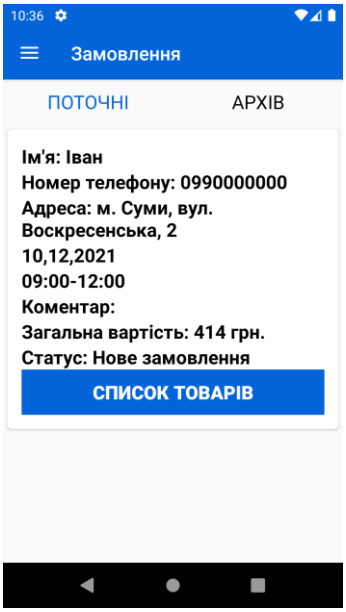


Рисунок 4.25 – Поточні замовлення користувача

Щоб переглянути список товарів замовлення, потрібно натиснути на відповідну кнопку (рис. 4.26).

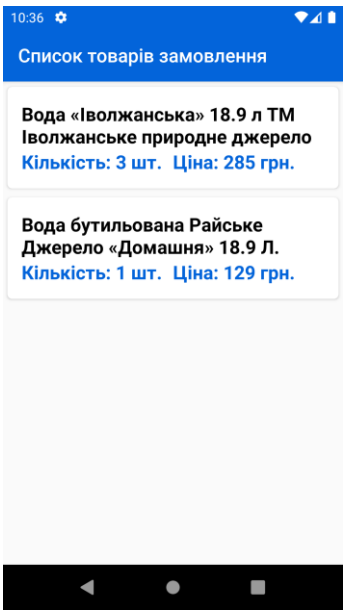


Рисунок 4.26 – Список товарів замовлення

Користувач має змогу відмінити власне замовлення, якщо затиснути на відповідному замовленні (рис. 4.27).

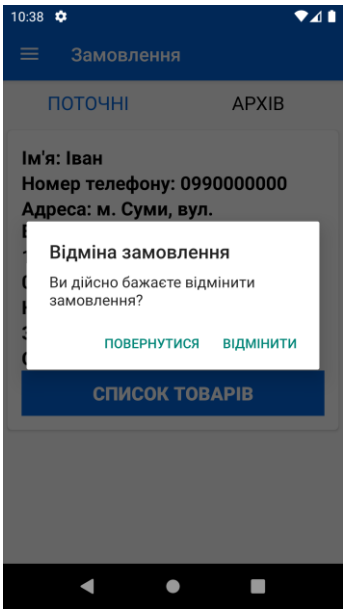


Рисунок 4.27 – Видалення замовлення

Переглянути архівні замовлення можна натиснувши на надпис “Архів” (рис. 4.28). Статус замовлення вказує на те, що це виконані чи відмінені замовлення.

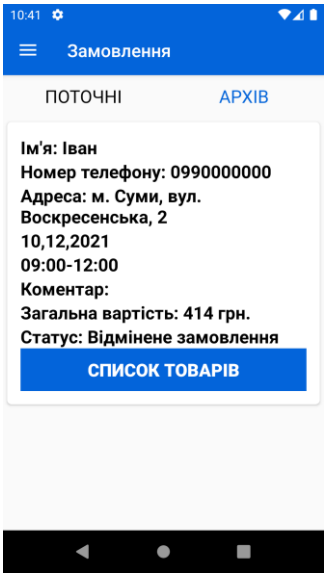


Рисунок 4.28 – Архівні замовлення

Для того, щоб вийти з облікового запису, користувачу потрібно натиснути на відповідний пункт бокового меню.

Після успішної авторизації адміністратору відкривається головний екран адміністративної панелі (рис. 4.29). Адміністратор має можливості додавати та редагувати товари, переглядати список замовлень та керувати кур’єрами.

Рисунок 4.29 – Адміністративна панель

Щоб додати товар, потрібно натиснути на відповідну кнопку. Після чого відкриється новий екран, де адміністратор повинен обрати зображення, ввести назву товару, його опис та ціну (рис. 4.30).

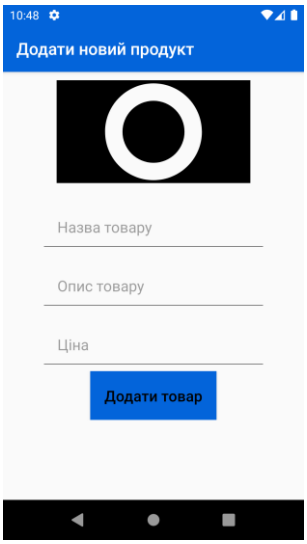


Рисунок 4.30 – Додавання товару

Вибір зображення відбувається шляхом відкриття галереї на мобільному телефоні (рис. 4.31), де адміністратор і обирає потрібне зображення.

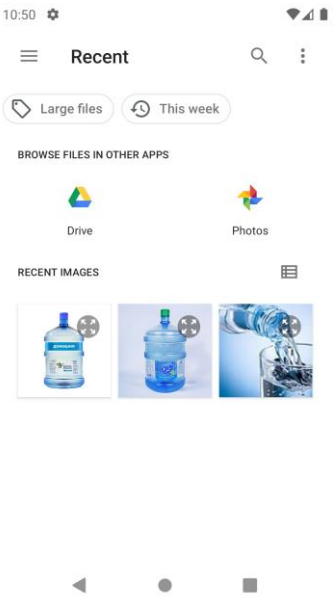


Рисунок 4.31 – Галерея

Після заповненя усіх полів форми необхідно натиснути на кнопку “Додати товар”, після чого почнеться процедура завантаження зображення та додавання запису в базу даних (рис. 4.32).

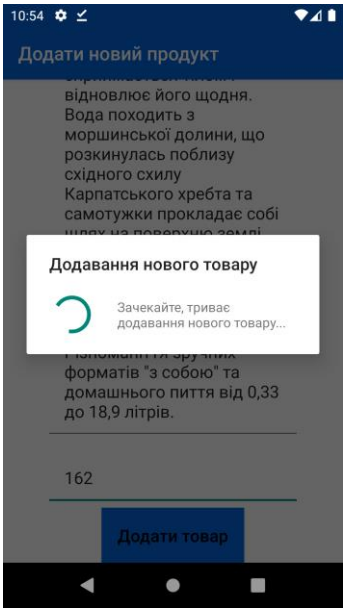


Рисунок 4.32 – Завантаження зображення

Доданий товар можна побачити в каталозі (рис. 4.33).

Рисунок 4.33 – Щойно доданий товар

Для редагування товару потрібно обрати його зі списку (рис. 4.34). Для полегшення вибору необхідного товару можна скористатися пошуком.

Рисунок 4.34 – Вибір товару для редагування

Натиснувши на потрібний товар, відкриється форма, де можна редагувати існуючу інформацію (рис. 4.35). Окрім цього, адміністратор має можливість видалити даний товар з каталогу, натиснувши на відповідну кнопку.

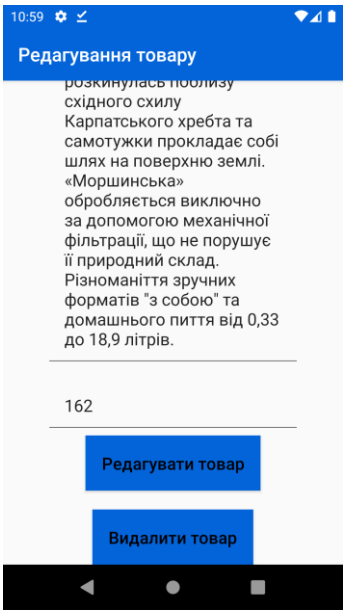


Рисунок 4.35 – Редагування товару

Адміністратор також має змогу переглядати поточний список замовлень усіх користувачів. Окрім цього він має можливість змінити статус замовлення на доставлений чи відмінений, тим самим перемістить замовлення в архів (рис. 4.36).

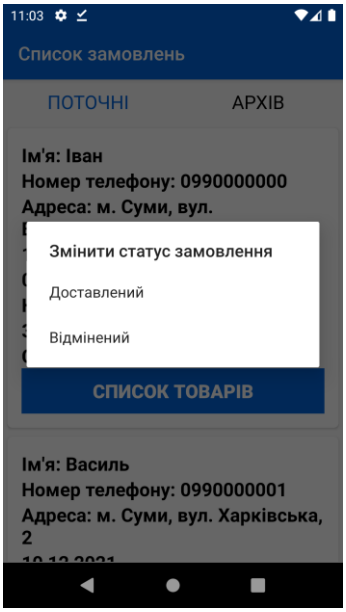


Рисунок 4.36 – Зміна статусу адміністратором

Щоб додати чи видалити кур’єра, потрібно ввести потрібний номер телефону облікового запису та натиснути на відповідну кнопку (рис. 4.37).

Рисунок 4.37 – Редагування списку кур’єрів

Після авторизації кур’єру доступний список замовлень на поточний день, який відсортований по годинам доставки (рис. 4.38).

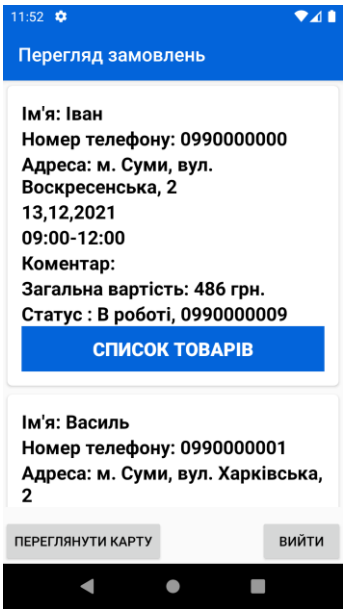


Рисунок 4.38 – Список поточних замовлень

Кур’єр може змінювати статус замовлення на три пункти: в роботі, доставлений та відмінений (рис. 4.39).

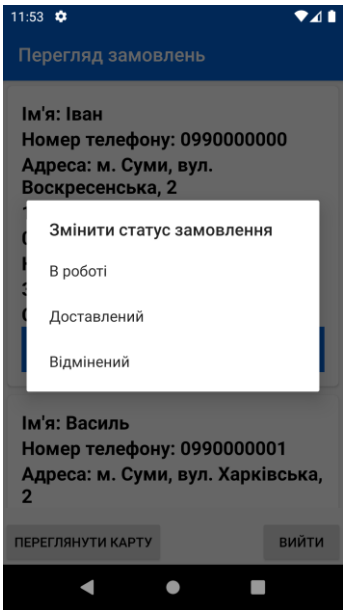


Рисунок 4.39 – Зміна статусу замовлення кур’єром

Якщо статус замовлення встановити як “В роботі”, то до самого статусу додається номер телефону кур’єра, який його виконує (рис. 4.40).

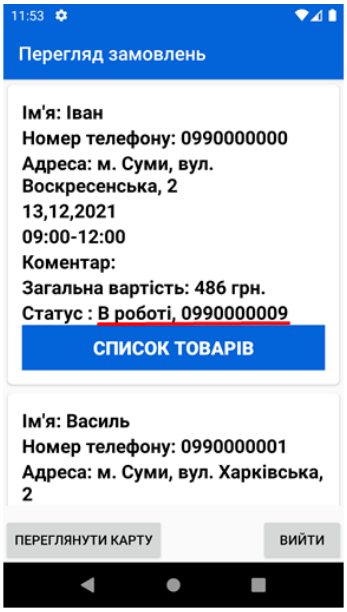


Рисунок 4.40 – Статус “В роботі”

Після натискання на замовлення відкриється екран з картою, де будуть відображені поточне місцезнаходження кур’єра та точку доставки. (рис. 4.41).

Рисунок 4.41 – Адреса доставки

Якщо натиснути на кнопку “Переглянути карту”, то на екрані з картою міста будуть відображатися усі точки доставки на поточний день. Різними кольорами показано замовлення на ранок (жовтий колір), день (помаранчевий) та вечір (червоний) (рис. 4.42).

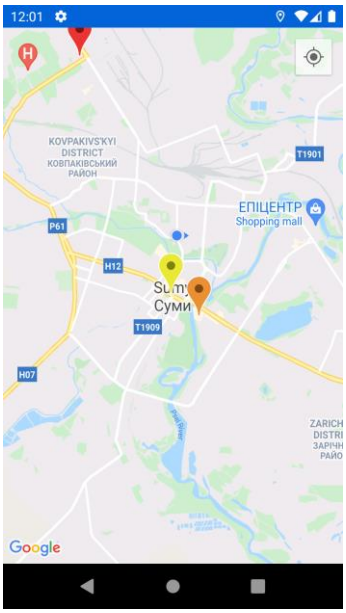


Рисунок 4.42 – Адреси доставок на день

Щоб дізнатися маршрут до точки доставки, потрібно обрати її на карті та натиснути на кнопку побудови маршруту (рис. 4.43).

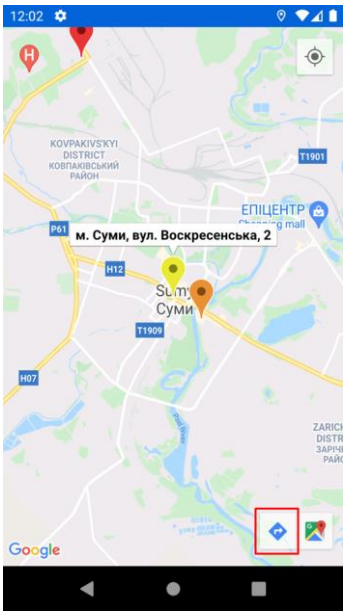


Рисунок 4.43 – Вибір точки доставки

Результатами цих дій буде маршрут до вибраної точки (рис. 4.44).

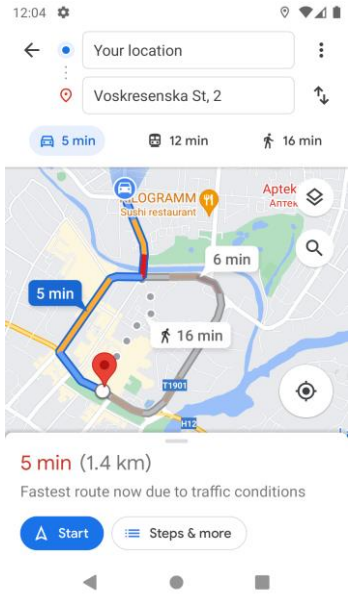


Рисунок 4.44 – Маршрут до точки доставки

ВИСНОВОК

У ході виконання кваліфікаційної роботи магістра було розроблено інформаційну систему організації замовлення бутильованої питної води.

Після проведення аналізу предметної області за темою роботи було визначено, що розробка та використання мобільних додатків організаціями є досить актуальними сьогодні. Також було проаналізовано мобільні програми-аналоги організації замовлення питної води та визначено їх функціонал. Результатом стала поставлена задача створити новий додаток організації замовлення питної води. В аналітичній частині даної науково-дослідної роботи були сформовані мета роботи, задачі для виконання та описані вимоги до продукту.

У частині роботи з планування робіт було ідентифіковано та деталізовано мету IT-проекту, розпланово зміст робіт та структуру виконавців у вигляді діаграм WBS та OBS, визначено строки на виконання задач у діаграмі Ганта та визначено можливі ризики проекту, їх ступінь впливу на план вирішення.

У проектній частині роботи було виконано структурно-функціональне моделювання, щоб визначити майбутній функціонал інформаційної системи, та визначено варіанти використання, котрі демонструють взаємодію системи з акторами.

У практичній частині роботи наведено архітектуру розроблюваної інформаційної системи. Також було описано файли реалізації мобільного додатку з прикладами роботи.

Використання мобільного додатку організації замовлення води дозволяє скоротити час користувача під час оформлення замовлення на доставку води та організації внутрішньої роботи компанії із замовленнями.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. InnverseTechnologies. What Importance of Mobile Applications in Everyday Life [Електронний ресурс] / InnverseTechnologies – Режим доступу до ресурсу: https://medium.com/@innversetech/what-importance-of-mobile-applications-in-everyday-life-dd2e27c6ee9e. (дата звернення 04.10.2021)

2. Zhao Z., Balagué C. Designing branded mobile apps: Fundamentals and recommendations //Business Horizons. – 2015. – Т. 58. – №. 3. – С. 305-315. (дата звернення 04.10.2021)

3. Baranova V. et al. Stochastic Frontier Analysis and Wavelet Ideology in the Study of Emergence of Threats in the Financial Markets //2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T). – IEEE, 2019. – С. 341-344. (дата звернення 05.10.2021)

4. Baranova V. et al. Wavelet Coherence as a Tool for Studying of Economic Dynamics in Infocommunication Systems //2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T). – IEEE, 2019. – С. 336-340. (дата звернення 06.10.2021)

5. Siegler, MG Analyst: There’s a great future in iPhone apps. [Електронний ресурс] / Venture Beat – Режим доступу до ресурсу: https://venturebeat.com/2008/06/11/analyst-theres-a-great-future-in-iphone-apps. (дата звернення 07.10.2021)

6. UAB The Future of Mobile Application [Електронний ресурс] / University of Alabama at Birmingham’s Business program – Режим доступу до ресурсу: https://businessdegrees.uab.edu/blog/the-future-of-mobile-application/. (дата звернення 08.10.2021)

7. Колесникова Т. А., Гарагуля І. О. Роль мобільних додатків в житті сучасної людини. – 2021. (дата звернення 09.10.2021)

8. De Kerviler G., Demoulin N. T. M., Zidda P. Adoption of in-store mobile payment: Are perceived risk and convenience the only drivers? //Journal of Retailing and Consumer Services. – 2016. – Т. 31. – С. 334-344. (дата звернення 10.10.2021)

9. Martins J. et al. How smartphone advertising influences consumers' purchase intention //Journal of Business Research. – 2019. – Т. 94. – С. 378-387. (дата звернення 11.10.2021)

10. Cortiñas M., Chocarro R., Elorz M. Omni-channel users and omni-channel customers: a segmentation analysis using distribution services //Spanish Journal of Marketing-ESIC. – 2019. (дата звернення 12.10.2021)

11. Orús C., Gurrea R., Ibáñez-Sánchez S. The impact of consumers' positive online recommendations on the omnichannel webrooming experience //Spanish Journal of Marketing-ESIC. – 2019. (дата звернення 13.10.2021)

12. Wang X. et al. Consumer participation in last-mile logistics service: an investigation on cognitions and affects //International Journal of Physical Distribution & Logistics Management. – 2019. (дата звернення 14.10.2021)

13. Chen M. C. et al. Ensuring the quality of e-shopping specialty foods through efficient logistics service //Trends in Food Science & Technology. – 2014. – Т. 35. – №. 1. – С. 69-82. (дата звернення 15.10.2021)

14. Visser J., Nemoto T., Browne M. Home delivery and the impacts on urban freight transport: A review //Procedia-social and behavioral sciences. – 2014. – Т. 125. – С. 15-27. (дата звернення 16.10.2021)

15. Mehmood S. M., Najmi A. Understanding the impact of service convenience on customer satisfaction in home delivery: Evidence from Pakistan //International Journal of Electronic Customer Relationship Management. – 2017. – Т. 11. – №. 1. – С. 23-43. (дата звернення 17.10.2021)

16. Alalwan A. A. Mobile food ordering apps: An empirical study of the factors affecting customer e-satisfaction and continued intention to reuse //International Journal of Information Management. – 2020. – Т. 50. – С. 28-44. (дата звернення 18.10.2021)

17. Belanche D., Flavián M., Pérez-Rueda A. Mobile apps use and wom in the food delivery sector: The role of planned behavior, perceived security and customer lifestyle compatibility //Sustainability. – 2020. – Т. 12. – №. 10. – С. 4275. (дата звернення 19.10.2021)

18. Ehmke J. F., Mattfeld D. C. Vehicle routing for attended home delivery in city logistics //Procedia-Social and Behavioral Sciences. – 2012. – Т. 39. – С. 622-632. (дата звернення 20.10.2021)

19. Bernal Jurado E. et al. Evaluation of corporate websites and their influence on the performance of olive oil companies //Sustainability. – 2018. – Т. 10. – №. 4. – С. 1274. (дата звернення 21.10.2021)

20. Top Benefits of Having a Bottled Water Delivery App [Електронний ресурс] – Режим доступу до ресурсу: https://www.onesneed.in/blog/top-benefits-of-having-a-bottled-water-delivery-app/. (дата звернення 02.10.2021)

21. 7 popular types of business applications for mobile phones [Електронний ресурс] – Режим доступу до ресурсу: https://www.outsource2india.com/software/mobile-applications/articles/business-applications-mobile-phones.asp. (дата звернення 03.10.2021)

22. Mobile App Development Process: Step-by-Step Guide [2021] [Електронний ресурс] – Режим доступу до ресурсу: https://www.invonto.com/insights/mobile-app-development-process/. (дата звернення 22.10.2021)

23. The Major Advantages of Android Studio App Development [Електронний ресурс] // Indianappdevelopers – Режим доступу до ресурсу: https://www.indianappdevelopers.com/blog/advantages-of-android-studio-app-development/. (дата звернення 25.10.2021)

24. Chatterjee N. et al. Real-time communication application based on android using Google firebase //Int. J. Adv. Res. Comput. Sci. Manag. Stud. – 2018. – Т. 6. – №. 4. (дата звернення 27.10.2021)

25. Godlevskyi M. D., Orlovskyi D. L., Kopp A. M. Structural analysis and optimization of IDEF0 functional business process models

//Радіоелектроніка, інформатика, управління. – 2018. – №. 3 (46). (дата звернення 09.11.2021)

26. Mora M. et al. Impacts of IDEF0-Based Models on the Usefulness, Learning, and Value Metrics of Scrum and XP Project Management Guides //Engineering Management Journal. – 2021. – С. 1-17. (дата звернення 09.11.2021)

27. Baghbani M. IDEF0 Modeling Standard: a tool for process map drawing under requirements of ISO 9001: 2015: a case study //Journal of Modern Processes in Manufacturing and Production. – 2019. – Т. 8. – №. 4. – С. 57-66. (дата звернення 11.11.2021)

28. Леоненков, А.В. Самоучитель UML 2 / А.В. Леоненков. – СПб.: БХВ - Петербург, 2007. – 576с. (дата звернення 12.11.2021)

29. Моделювання програмного забезпечення, [Електронний ресурс] – режим доступу: http://elartu.tntu.edu.ua/bitstream/123456789/17796/1/Моделювання%20програмного%20забезпечення.pdf (дата звернення 15.11.2021)

30. Lang J., Spišák D. Activity Diagram as an Orientation Catalyst within Source Code //Acta Polytechnica Hungarica. – 2021. – Т. 18. – №. 3. – С. 127-146. (дата звернення 17.11.2021)

31. Stelzle B., Noennig J. R. A database for participation methods in urban development //Procedia computer science. – 2017. – Т. 112. – С. 2416-2425. (дата звернення 19.11.2021)

32. JSON: основи використання [Електронний ресурс] – Режим доступу до ресурсу: https://ruseller.com/lessons.php?rub=28&id=1212. (дата звернення 20.11.2021)

33. Lou T. et al. A comparison of Android Native App Architecture MVC, MVP and MVVM //Eindhoven University of Technology. – 2016. (дата звернення 25.11.2021)

34. C. Anderson, "The Model-View-ViewModel (MVVM) Design Pattern," in Pro Business Applications with Silverlight 5, Berkeley, Apress, 2012, pp. 461--499. (дата звернення 28.11.2021)

Додаток А. Планування робіт

А.1 Ідентифікація мети ІТ-проекту

Метою кваліфікаційної роботи магістра є розробка інформаційної системи у вигляді мобільного додатку, який буде забезпечувати належну організацію замовлення бутильованої питної води за рахунок скорочення часу при оформленні замовлення за допомогою розробленого продукту та організації доставки цих самих замовлень постачальником. Використовуючи мобільний додаток, клієнт може замовити товари, попередньо прочитавши інформацію про них, не гаючи час на телефонний дзвінок у будь-який час. Окрім цього, особа, що здійснює доставку, віддалена від комп'ютера, тобто, функціонал повинен бути доступний з будь-якого місця, тому потрібно зробити мобільний додаток.

Для конкретизації мети проекту було використано технологію S.M.A.R.T. Ця технологія дає критерії, якими можна орієнтуватися при постановці цілей в управлінні проектом, управлінні ефективністю співробітників та особистому розвитку.

Результати деталізації методом S.M.A.R.T розміщені у табл. А.1.

Таблиця А.1 – Деталізація мети проекту методом S.M.A.R.T

Specific (конкретна)	Розробити інформаційну систему організації замовлення бутильованої питної води.
Measurable (вимірювання)	Оскільки даний проект не є комерційним, то результатом його роботи є оцінка замовника.

Продовження таблиці А.1 – Деталізація мети проекту методом S.M.A.R.T

Achievable (досяжна, узгоджена)	Ціль даного проекту вважається досяжною, так як розробник володіє усіма необхідними знаннями для роботи в середовищі розроки Android Studio на мові програмування Java, які використовуються для розробки даного мобільного додатку.
Relevant (реалістична)	У наявності є всі необхідні технічні та програмні засоби. Розробник достатньо кваліфікований для виконання поставлених задач.
Time-framed (обмежена в часі)	Ціль має часове обмеження, оскільки інформаційна система розроблюється з обмеженням у часі на основі сформованого календарного плану.

А.2 Планування змісту структури робіт ІТ-проекту

Для планування змісту структури робіт використовують WBS (Work Break Structure). Це графічне представлення згрупованих елементів проекту у вигляді пакета робіт, які ієрархічно пов’язані з продуктом проекту. На першому рівні WBS фіксується продукт проекту. Він повинен відповідати продукту проекту. Наступний ІІ рівень відповідає діям або основним заходам для досягнення продукту проекту.

Потім триває розбивка цих дій доти, поки не відбувається виконання дій елементарних робіт. Елементарні роботи – це роботи, які мають один чіткий результат, який використовується при прийнятті цієї роботи; на які призначений один конкретний відповідальний; на неї можна обчислити витрати праці і тривалість виконання. Зазвичай декомпозиція завершується тоді, коли для розкриття змісту потрібні вузькі фахівці, що знають

технологічні особливості їх виконання. Діаграма WBS представлена на рис. А.1.

Після побудови WBS розробляють організаційну структуру виконавців.

OBS – організаційна структура виконавців (організацій) проекту. Визначається за переліком пакетів робіт нижнього рівня кожної гілки WBS-структури. Представляється відповідальними (відповідальні – це не обов'язково керівники організацій(відділів), а ті люди які безпосередньо організують виконання робіт) за виконання пакетів робіт.

Організаційна структура представляє собою графічне відображення учасників проекту та їх відповідальних осіб, які задіяні в реалізації проекту. На верхньому рівні OBS розташована команда проекту. На наступному рівні фіксуються виконавці: організації, відділи тощо. Потім, рівнем нижче, для кожного виконавця вказують прізвища конкретних осіб, які будуть відповідати за виконання елементарних робіт WBS. Потрібно пам'ятати, що відповідальні – це не обов'язково керівники, а ті співробітники, які безпосередньо організовують і відповідають у виконавця за виконання елементарної роботи, зазначеної у WBS. Для них ця елементарна робота також є проектом (у порівнянні з загальним проектом). Для себе вони також можуть побудувати WBS- структуру й застосовувати інші інструменти планування. Саме на цьому рівні закладається певна якість майбутнього продукту проекту.

Діаграма OBS представлена на рис. А.2. Список виконавців, що функціонують в проекті знаходиться в табл. А.2.

Таблиця А.2 – Список виконавців проекту

Ім'я	Роль	Проектна роль
Андрусишин І.К.	Виконавець	Виконує розробку інформаційної системи.
Парфененко Ю.В.	Керівник проекту	Формує завдання на розробку проекту.

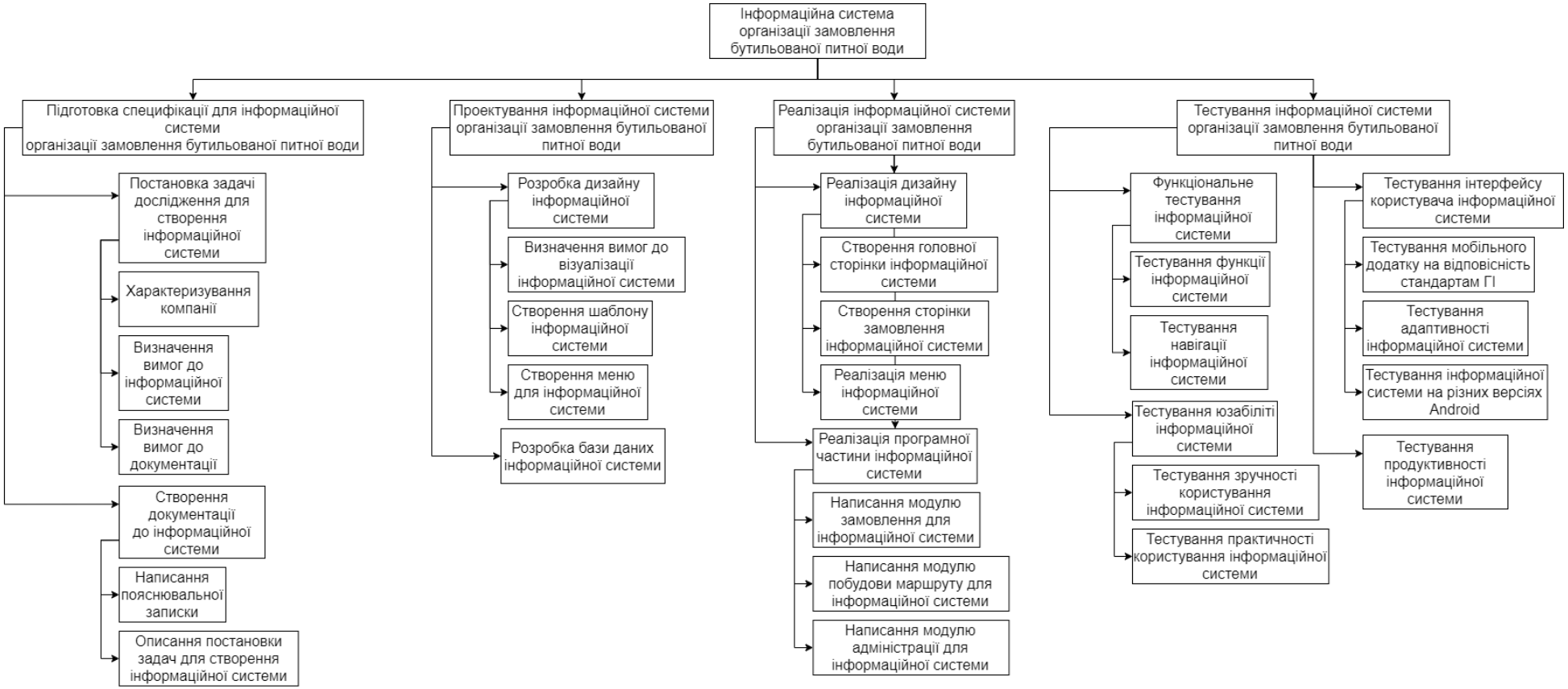


Рисунок А.1 – WBS. Структура робіт проекту

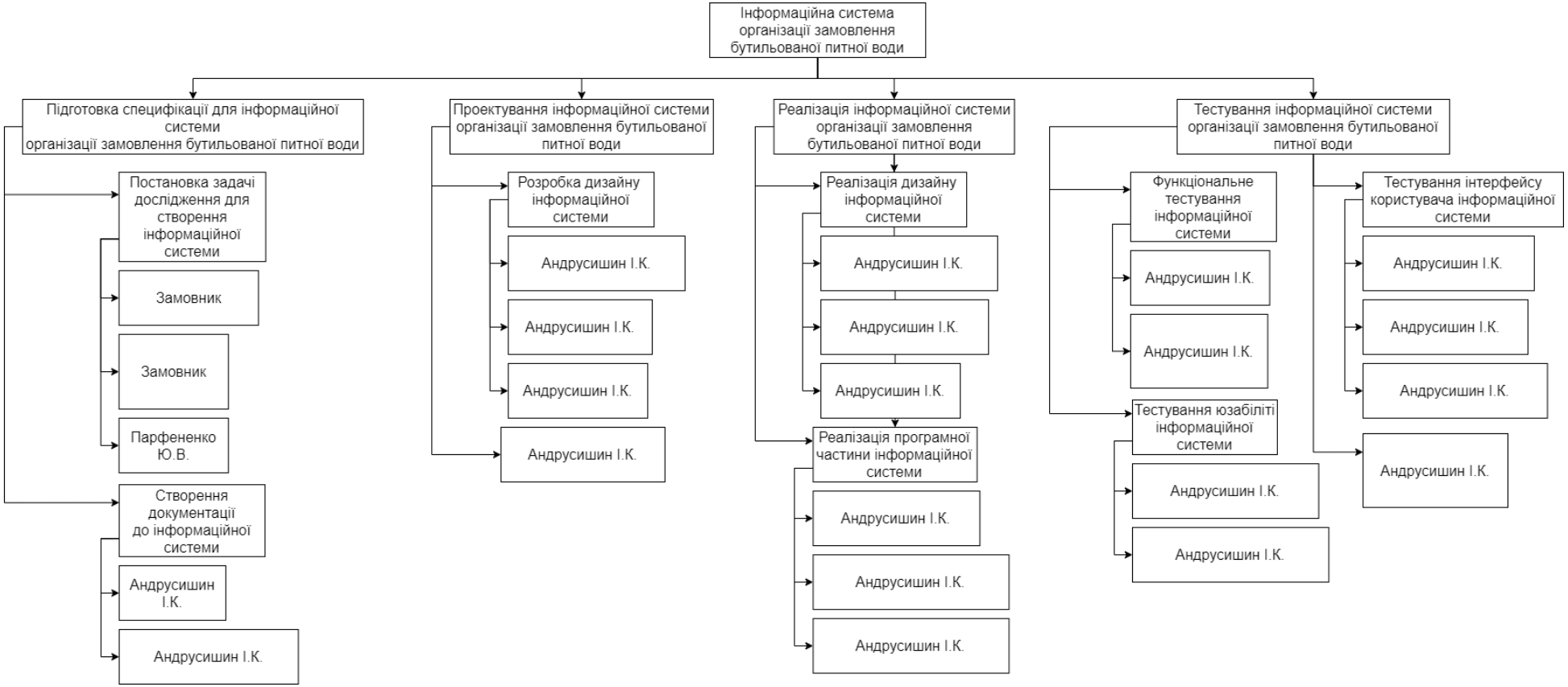
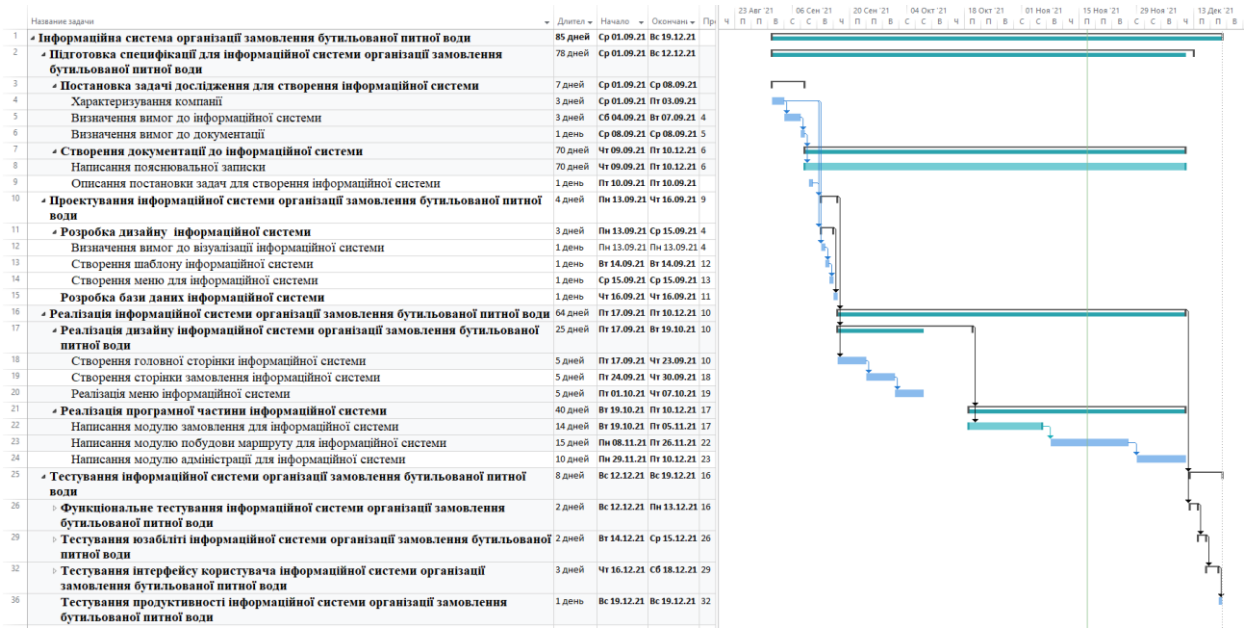


Рисунок А.2 – OBS структура

А.3 Побудова календарного графіку виконання ІТ - проекту

Діаграма Ганта – це інструмент управління проектами, що допомагає планувати проекти всіх розмірів. Терміни та завдання управління проектами перетворюються на горизонтальну смужкову діаграму із зазначенням дати початку та закінчення, а також залежностей, планування та строків, у тому числі, скільки завдань виконано на етапі та хто є власником завдання. Це корисно для відстеження завдань, коли існує велика команда та декілька зацікавлених сторін, коли область застосування змінюється.

Діаграма Ганта представлена на рис. А.3.



	Название задачи	Длител	Начало	Окончани
1	**Інформаційна система організації замовлення бутильованої питної води**	**85 дней**	**Ср 01.09.21**	**Вс 19.12.21**
2	**Підготовка специфікації для інформаційної системи організації замовлення бутильованої питної води**	78 дней	Ср 01.09.21	Вс 12.12.21
3	**Постановка задачі дослідження для створення інформаційної системи**	7 дней	Ср 01.09.21	Ср 08.09.21
4	Характеризування компанії	3 дней	Ср 01.09.21	Пт 03.09.21
5	Визначення вимог до інформаційної системи	3 дней	Сб 04.09.21	Вт 07.09.21
6	Визначення вимог до документації	1 день	Ср 08.09.21	Ср 08.09.21
7	**Створення документації до інформаційної системи**	70 дней	Чт 09.09.21	Пт 10.12.21
8	Написання пояснювальної записки	70 дней	Чт 09.09.21	Пт 10.12.21
9	Описання постановки задач для створення інформаційної системи	1 день	Пт 10.09.21	Пт 10.09.21
10	**Проектування інформаційної системи організації замовлення бутильованої питної води**	4 дней	Пн 13.09.21	Чт 16.09.21
11	**Розробка дизайну інформаційної системи**	3 дней	Пн 13.09.21	Ср 15.09.21
12	Визначення вимог до візуалізації інформаційної системи	1 день	Пн 13.09.21	Пн 13.09.21
13	Створення шаблону інформаційної системи	1 день	Вт 14.09.21	Вт 14.09.21
14	Створення меню для інформаційної системи	1 день	Ср 15.09.21	Ср 15.09.21
15	**Розробка бази даних інформаційної системи**	1 день	Чт 16.09.21	Чт 16.09.21
16	**Реалізація інформаційної системи організації замовлення бутильованої питної води**	64 дней	Пт 17.09.21	Пт 10.12.21
17	**Реалізація дизайну інформаційної системи організації замовлення бутильованої питної води**	25 дней	Пт 17.09.21	Вт 19.10.21
18	Створення головної сторінки інформаційної системи	5 дней	Пт 17.09.21	Чт 23.09.21
19	Створення сторінки замовлення інформаційної системи	5 дней	Пт 24.09.21	Чт 30.09.21
20	Реалізація меню інформаційної системи	5 дней	Пт 01.10.21	Чт 07.10.21
21	**Реалізація програмної частини інформаційної системи**	40 дней	Вт 19.10.21	Пт 10.12.21
22	Написання модулю замовлення для інформаційної системи	14 дней	Вт 19.10.21	Пт 05.11.21
23	Написання модулю побудови маршруту для інформаційної системи	15 дней	Пн 08.11.21	Пт 26.11.21
24	Написання модулю адміністрації для інформаційної системи	10 дней	Пн 29.11.21	Пт 10.12.21
25	**Тестування інформаційної системи організації замовлення бутильованої питної води**	8 дней	Вс 12.12.21	Вс 19.12.21
26	**Функціональне тестування інформаційної системи організації замовлення бутильованої питної води**	2 дней	Вс 12.12.21	Пн 13.12.21
29	**Тестування юзабіліті інформаційної системи організації замовлення бутильованої питної води**	2 дней	Вт 14.12.21	Ср 15.12.21
32	**Тестування інтерфейсу користувача інформаційної системи організації замовлення бутильованої питної води**	3 дней	Чт 16.12.21	Сб 18.12.21
36	**Тестування продуктивності інформаційної системи організації замовлення бутильованої питної води**	1 день	Вс 19.12.21	Вс 19.12.21

Рисунок А.3 – Діаграма Ганта

А.4 Управління ризиками проекту

Управління ризиками проекту включає процеси, пов’язані із здійсненням планування управління ризиками, виявленням, аналізом, реагуванням, а також моніторингом і контролем ризиків в проекті.

Процес управління ризиками включає наступні етапи:

1) планування управління ризиками;
2) ідентифікація ризиків;
3) проведення якісного аналізу ризиків;
4) проведення кількісного аналізу ризиків;
5) планування реагування на відомі ризики;
6) моніторинг та контроль ризиків.

Ризики проекту представлені у таблиці А.3

Таблиця А.3 – Ризики проекту

№	Назва	Опис	План вирішення ризиків	Вірогідність виникнення	Ступінь впливу
1	Технічні проблеми	Виникнення технічних неполадок, що можуть призвести до пошкодження чи видалення файлів проекту.	Збереження та оновлення копії проекту на хмарному середовищі.	Низька	Висока
2	Важкість інтеграції	Ризик виникнення різних проблем в процесі інтеграції модулів адміністрування та побудови маршруту.	Використання модулів з легкою інтеграцією.	Середня	Середнія
3	Низька продуктивність	Нерівномірна продуктивність, активізація тільки ближче до кінця терміну здачі проекту, а в решту часу праця в півсили.	Використання гнучкої методології розробки.	Висока	Висока

Додаток Б. Програмний код мобільного додатку

Код файлу HomeActivity.java

```
package com.example.water;

import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.net.Uri;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.LayoutInflater;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.view.Menu;
import android.view.ViewGroup;
import android.widget.SearchView;
import android.widget.TextView;
import android.widget.Toast;

import com.example.water.Model.Orders;
import com.example.water.Model.Products;
import com.example.water.Model.Users;
import com.example.water.Prevalent.Prevalent;
import com.example.water.ViewHolder.ProductViewHolder;
import com.firebase.ui.database.FirebaseRecyclerAdapter;
import com.firebase.ui.database.FirebaseRecyclerOptions;
import com.google.android.material.floatingactionbutton.FloatingActionButton;
import com.google.android.material.snackbar.Snackbar;
import com.google.android.material.navigation.NavigationView;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;
import com.squareup.picasso.Picasso;

import androidx.annotation.NonNull;
```

```
import androidx.appcompat.app.ActionBarDrawerToggle;
import androidx.core.view.GravityCompat;
import androidx.navigation.NavController;
import androidx.navigation.Navigation;
import androidx.navigation.ui.AppBarConfiguration;
import androidx.navigation.ui.NavigationUI;
import androidx.drawerlayout.widget.DrawerLayout;
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import io.paperdb.Paper;

public class HomeActivity extends AppCompatActivity implements
NavigationView.OnNavigationItemSelectedListener{

    private DatabaseReference ProductRef;
    private DatabaseReference PRef;
    private RecyclerView recyclerView;
    private NavigationView navigationView;
    private NavController navController;
    RecyclerView.LayoutManager layoutManager;
    private ActionBarDrawerToggle toggle;
    private Menu menu;
    private String admin = "";

    private AppBarConfiguration mAppBarConfiguration;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_home);
        setTitle("Оберіть товар для зміни");

        Intent intent = getIntent();
        Bundle bundle = intent.getExtras();
        if (bundle != null){
            admin = getIntent().getExtras().get("Admin").toString();
        }
```

```
        ProductRef = FirebaseDatabase.getInstance().getReferenceFromUrl("https://water-f1ab5-default-
rtdb.firebaseio.com/").child("Products");
        recyclerView = findViewById(R.id.recycler_menu);
        recyclerView.setHasFixedSize(true);
        layoutManager = new LinearLayoutManager(this);
        recyclerView.setLayoutManager(layoutManager);
        Paper.init(this);
        Toolbar toolbar = findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);
        FloatingActionButton fab = findViewById(R.id.fab);
        navigationView = findViewById(R.id.nav_view);
        fab.setVisibility(View.GONE);
        navigationView.setVisibility(View.GONE);
        DrawerLayout drawer = findViewById(R.id.drawer_layout);
        drawer = findViewById(R.id.drawer_layout);
        drawer.setDrawerLockMode(DrawerLayout.LOCK_MODE_LOCKED_CLOSED);

        if (!admin.equals("Admin")){
            setTitle("Каталог");
            fab.setVisibility(View.VISIBLE);
            navigationView.setVisibility(View.VISIBLE);
            fab.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View view) {
                    Intent intent;
                    try{
                        Prevalent.currentOnlineUser.getName();
                        intent = new Intent(HomeActivity.this, CartActivity.class);
                        startActivity(intent);
                    } catch (Exception e){
                        intent = new Intent(HomeActivity.this, AuthActivity.class);
                        startActivity(intent);
                    }
                }
            });
            navigationView = findViewById(R.id.nav_view);
            navigationView.setNavigationItemSelectedListener(this);
            drawer.setDrawerLockMode(DrawerLayout.LOCK_MODE_UNLOCKED);

        toggle = new ActionBarDrawerToggle(this, drawer, toolbar, R.string.navigation_drawer_open,
R.string.navigation_drawer_close){
```

```
            @Override
            public void onDrawerOpened(View drawerView) {
                super.onDrawerOpened(drawerView);
                try{
                    TextView userNameTextView = findViewById(R.id.user_profile_name);
                    userNameTextView.setText(Prevalent.currentOnlineUser.getName());
                } catch (Exception e){
                    TextView userNameTextView = findViewById(R.id.user_profile_name);
                    userNameTextView.setText("Доставка води");
                    //Toast.makeText(getApplicationContext(), "Помилка завантаження!",
Toast.LENGTH_SHORT).show();
                }
            }
        };

        toggle.syncState();

        drawer.addDrawerListener(toggle);
        menu = navigationView.getMenu();
    }

    navigationView = findViewById(R.id.nav_view);
    mAppBarConfiguration = new AppBarConfiguration.Builder(
            R.id.nav_cart, R.id.nav_orders, R.id.nav_settings, R.id.nav_logout)
            .setDrawerLayout(drawer)
            .build();

    NavigationView navigationView = (NavigationView) findViewById(R.id.nav_view);
    navigationView.setNavigationItemSelectedListener(this);

    View headerView = navigationView.getHeaderView(0);
    TextView userNameTextView = headerView.findViewById(R.id.user_profile_name);
    userNameTextView.setText(Prevalent.currentOnlineUser.getName());

}



@Override
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.home, menu);
    MenuItem item=menu.findItem(R.id.searchbtn);
```

```
        SearchView searchView=(SearchView) item.getActionView();
        searchView.requestFocus();
        searchView.setOnQueryTextListener(new SearchView.OnQueryTextListener() {
            @Override
            public boolean onQueryTextSubmit(String query) {
                process_search(query);
                return false;
            }

            @Override
            public boolean onQueryTextChange(String newText) {
                process_search(newText);
                return false;
            }
        });
        return true;
    }

    private void process_search(String s){

        if (s.isEmpty()){
            onStart();
        }
        else {
        FirebaseRecyclerOptions<Products> options = new
FirebaseRecyclerOptions.Builder<Products>().setQuery(FirebaseDatabase.getInstance().getReference().child("Prod
ucts").orderByChild("goods_name").startAt(s).endAt(s+"\uf8ff"), Products.class).build();
        FirebaseRecyclerAdapter<Products, ProductViewHolder> adapter = new FirebaseRecyclerAdapter<Products,
ProductViewHolder>(options) {
            @Override
            protected void onBindViewHolder(@NonNull ProductViewHolder productViewHolder, int i, @NonNull
Products products) {
                productViewHolder.product_name.setText(products.getGoods_name());
                productViewHolder.product_price.setText("Ціна : " + products.getGoods_price() + " грн.");
                Picasso.get().load(products.getGoods_image()).into(productViewHolder.product_image);

                productViewHolder.itemView.setOnClickListener(new View.OnClickListener() {
                    @Override
                    public void onClick(View v) {
                        if (admin.equals("Admin")){
                            Intent intent = new Intent(HomeActivity.this, AdminManageActivity.class);
                            intent.putExtra("goods_id", products.getGoods_id());
```

```
                    startActivity(intent);
                }
                else {

                    Intent intent = new Intent(HomeActivity.this, ProductDetailsActivity.class);
                    intent.putExtra("goods_id", products.getGoods_id());
                    startActivity(intent);
                }


            }
        });
    }

    @NonNull
    @Override
    public ProductViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
        View view = LayoutInflater.from(parent.getContext()).inflate(R.layout.product_items_layout, parent,
false);
        ProductViewHolder holder = new ProductViewHolder(view);
        return holder;
    }
  };
  recyclerView.setAdapter(adapter);
  adapter.startListening();
}
}


@Override
public boolean onOptionsItemSelected(MenuItem item)
{
  int id = item.getItemId();
  return super.onOptionsItemSelected(item);
}

@Override
protected void onStart() {
  super.onStart();
  FirebaseRecyclerOptions<Products> options = new
FirebaseRecyclerOptions.Builder<Products>().setQuery(ProductRef, Products.class).build();
```

```
        FirebaseRecyclerAdapter<Products, ProductViewHolder> adapter = new FirebaseRecyclerAdapter<Products,
ProductViewHolder>(options) {
            @Override
            protected void onBindViewHolder(@NonNull ProductViewHolder productViewHolder, int i, @NonNull
Products products) {
                productViewHolder.product_name.setText(products.getGoods_name());
                productViewHolder.product_price.setText("Ціна : " + products.getGoods_price() + " грн.");
                Picasso.get().load(products.getGoods_image()).into(productViewHolder.product_image);

                productViewHolder.itemView.setOnClickListener(new View.OnClickListener() {
                    @Override
                    public void onClick(View v) {
                        if (admin.equals("Admin")){
                            Intent intent = new Intent(HomeActivity.this, AdminManageActivity.class);
                            intent.putExtra("goods_id", products.getGoods_id());
                            startActivity(intent);
                        }
                        else {

                            Intent intent = new Intent(HomeActivity.this, ProductDetailsActivity.class);
                            intent.putExtra("goods_id", products.getGoods_id());
                            startActivity(intent);
                        }
                    }
                });
            }

            @NonNull
            @Override
            public ProductViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
                View view = LayoutInflater.from(parent.getContext()).inflate(R.layout.product_items_layout, parent,
false);
                ProductViewHolder holder = new ProductViewHolder(view);
                return holder;
            }
        };
        recyclerView.setAdapter(adapter);
        adapter.startListening();
    }


    @SuppressWarnings("StatementWithEmptyBody")
```

```
@Override
public boolean onNavigationItemSelected(MenuItem item)
{
    int id = item.getItemId();

    if (id == R.id.nav_cart)
    {
       try{
          Prevalent.currentOnlineUser.getName();
          Intent intent = new Intent(HomeActivity.this, CartActivity.class);
          startActivity(intent);
       } catch (Exception e){
          Intent intent = new Intent(HomeActivity.this, AuthActivity.class);
          startActivity(intent);
       }
    }

    else if (id == R.id.nav_connect)
    {
       Intent intent = new Intent(Intent.ACTION_DIAL, Uri.fromParts("tel", "+380991234567", null));
       startActivity(intent);
    }
    else if (id == R.id.nav_orders)
    {
       try{
          Prevalent.currentOnlineUser.getName();
          Intent intent = new Intent(HomeActivity.this, UserOrdersActivity.class);
          startActivity(intent);
       } catch (Exception e){
          Intent intent = new Intent(HomeActivity.this, AuthActivity.class);
          startActivity(intent);
       }
    }
    else if (id == R.id.nav_settings)
    {
       try{
          Prevalent.currentOnlineUser.getName();
          Intent intent = new Intent(HomeActivity.this, SettingsActivity.class);
          startActivity(intent);
       } catch (Exception e){
          Intent intent = new Intent(HomeActivity.this, AuthActivity.class);
          startActivity(intent);
```

```
            }
        }
        else if (id == R.id.nav_logout)
        {
            Paper.book().delete(Prevalent.UserPhoneKey);
            Paper.book().delete(Prevalent.UserPasswordKey);
            Paper.book().destroy();
            System.exit(0);
            finish();

            Intent intent = new Intent(HomeActivity.this, HomeActivity.class);
            intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK | Intent.FLAG_ACTIVITY_CLEAR_TASK);
            finish();
            startActivity(intent);
        }

        DrawerLayout drawer = (DrawerLayout) findViewById(R.id.drawer_layout);
        drawer.closeDrawer(GravityCompat.START);
    return true;
  }
}
```