МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет кораблебудування
імені адмірала Макарова
Херсонський навчально-науковий інститут

Кафедра інформаційних технологій та фізико-математичних дисциплін

«Допущений до захисту»
Завідувач кафедри
д.т.н., доцент Гучек П.Й.
«22» червня 2024 р.

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи на здобуття ступеня вищої освіти «бакалавр»

на тему: “Розробка мобільного додатку доставки піци «PizzaShop»”

Виконав: студент IV курсу, групи 4157
спеціальності
121 - Інженерія програмного забезпечення
(шифр і назва спеціальності)
освітньої програми
Інженерія програмного забезпечення
(назва освітньої програми)
Бріттнер Р-Н.О.
(прізвище та ініціали)

Керівник Латанська Л.О.
(прізвище та ініціали)

Рецензент Макарова Л.М.
(прізвище та ініціали)

м. Миколаїв-Херсон - 2024 р.

Форма № Н-9.02

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ КОРАБЛЕБУДУВАННЯ
імені адмірала Макарова
ХЕРСОНСЬКИЙ НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ

Кафедра інформаційних технологій та фізико-математичних дисциплін
Галузь знань 12 "Інформаційні технології"
(шифр і назва)
Спеціальність 121 "Інженерія програмного забезпечення"
(шифр і назва)
Освітня програма "Інженерія програмного забезпечення"
(назва)

«ЗАТВЕРДЖУЮ»
Гарант освітньої програми
_____ Літвінова М.Б.
" 17 " березня 2024р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ
на здобуття ступеня вищої освіти «бакалавр»

Студенту Бріттнеру Райнеру-Нікіті Олександровичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи "Розробка мобільного додатку доставки піци «PizzaShop»"

Керівник роботи Латанська Людмила Олексіївна, к.ф.-м.н., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
Затверджені розпорядженням по інституту від "16" березня 2024 року № 04
2. Строк подання студентом роботи 03.06.2024 р.
3. Вихідні дані до роботи

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити):
4.1 Титульний аркуш, ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ (ДИПЛОМНУ) РОБОТУ, АНОТАЦІЯ (українською, російською, англійською), ЗМІСТ, ПЕРЕЛІК УМОВНИХ ОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ ТА ТЕРМІНІВ (при необхідності).
4.2 ВСТУП
4.3 ОПИС ПРЕДМЕТНОЇ ГАЛУЗІ, ПОСТАНОВКА ЗАДАЧІ
4.4 ПРОЕКТ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ (ПЗ)(зовнішні специфікації ПЗ, стани програми при виконанні, структура програми, структура даних, потоки даних та управління, проект ПЗ на рівні програмних модулів, випробування ПЗ)
4.5 РЕЗУЛЬТАТИ РОЗРОБКИ
4.6 РОЗДІЛ З ОХОРОНИ ПРАЦІ (ОХОРОНИ ОТОЧУЮЧОГО СЕРЕДОВИЩА)

ОХОРОНА ПРАЦІ
4.7 ВИСНОВКИ
4.8 СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ
4.9 ДОДАТКИ (технічне завдання, опис програми, текст програми, програма та методика випробувань ПЗ, інструкція користувача)

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

5.1	Діаграма варіантів використання до мобільного додатку «PizzaShop»
5.2	Концептуальна модель даних для системи Data Base
5.3	Діаграма класів системи Admin System
5.4	Діаграма класів системи User System
5.5	Діаграма послідовності дій користувачів
5.6	Логічна модель даних системи Admin System
5.7	Логічна модель даних системи User System
5.8	Діаграма компонентів

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
4.1-4.3	Дудченко О.М., декан		
4.4-4.5	Латанська Л.О., доцент		
4.6	Щедролосєв О.В., професор		

7. Дата видачі завдання 24 березня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Строк виконання етапів роботи	Примітка
1.	Опис та аналіз предметної галузі	28.03.2024 р.	
2.	Постановка та формалізація задачі	06.04.2024 р.	
3.	Ескізний проект програмного забезпечення	13.04.2024 р.	
4.	Технічний проект програмного забезпечення	20.04.2024 р.	
5.	Робочий проект програмного забезпечення	27.04.2024 р.	
6.	Випробування програмного забезпечення	05.05.2024 р.	
7.	Розробка робочої документації	11.05.2024 р.	
8.	Виконання графічних документів	13.05.2024 р.	
9.	Вирішення питань охорони праці	18.05.2024 р.	
10.	Оформлення пояснювальної записки	25.05.2024 р.	
11.	Подання кваліфікаційної роботи на попередній захист	02.06.2024 р.	
12.	Подання кваліфікаційної роботи рецензенту	06.06.2024 р.	
13.	Подання на кафедру ІТ та ФМД тексту остаточного варіанту роботи, підписаного її керівником, разом з заявою щодо самостійності виконання роботи та ідентичності друкованої та електронної версії роботи	08.06.2024 р.	
14.	Подання на кафедру ІТ та ФМД електронних версії наступних документів у форматі pdf: кваліфікаційної (дипломної) роботи; файлу-опису кваліфікаційної роботи (згідно Додатку до наказу ректора НУК від 19.05.2020 р. за №287-уч); презентації доповіді; письмового відгуку та рецензії на кваліфікаційну роботу	14.06.2024 р.	
15	Подання на кафедру ІТ ФМД письмового відгуку та рецензії на кваліфікаційну роботу	22.06.2024 р.	

Студент _____ Бріттнер Р-Н.О.
Підпис Прізвище та ініціали

Керівник роботи _____ Латанська Л.О.
Підпис Прізвище та ініціали

АНОТАЦІЯ

У цій кваліфікаційній роботі представлено дослідження та розробку мобільного додатку для доставки піци, спрямованого на покращення користувацького досвіду та оптимізацію процесу замовлення. Основна мета роботи полягала у створенні зручного, інтуїтивно зрозумілого та функціонального інструменту, який дозволить користувачам швидко і легко замовляти піцу з доставкою додому або в офіс.

У процесі розробки використана така мова програмування, як Swift для iOS а також серверна технологія, така як Firebase від Google. Особливу увагу приділено безпеці даних користувачів.

Результати роботи демонструють успішне створення додатка, який з часом доопрацювань буде відповідати сучасним стандартам і вимогам ринку. Проведені тестування підтвердили зручність та надійність розробленого рішення. Додаток може бути впроваджений у роботу будь-якої компанії, яка займається доставкою піци, що дозволить значно підвищити рівень обслуговування клієнтів і збільшити обсяг продажів.

Крім того, було реалізовано базовий адміністративний функціонал і інтерфейс, що дозволяє переглядати, керувати станами замовлення та додавати нові продукції.

Програмне забезпечення надає змогу зі сторони кліента:

- Створити обліковий запис.
- Налаштувати дані користувача.
- Зробити замовлення.
- Слідкувати за етапами замовлення.

Робота викладена українською мовою і виконана на 123 сторінках, містить 43 рисунки, 27 таблиць, 5 додатків. Список використаних джерел містить 13 найменувань.

Ключові слова: мобільний додаток, доставки піци, Swift, Firebase, ескізний проект, технічний проект, робочий проект.

ABSTRACT

This thesis presents the research and development of a mobile application for pizza delivery aimed at improving user experience and optimizing the ordering process. The main goal of the project was to create a convenient, intuitive, and functional tool that allows users to quickly and easily order pizza for delivery to their home or office.

In the development process, programming languages such as Swift for iOS and server technology like Firebase by Google were used. Special attention was paid to the security of user data.

The results of the work demonstrate the successful creation of an application that, with further refinements, will meet modern standards and market requirements. The tests conducted confirmed the convenience and reliability of the developed solution. The application can be implemented by any company engaged in pizza delivery, which will significantly improve the level of customer service and increase sales volume.

Additionally, basic administrative functionality and interface have been implemented, allowing for order management and the addition of new products.

The software provides the following capabilities for the client:

- Create an account.
- Configure user data.
- Place an order.
- Track the stages of delivery

The work is presented in Ukrainian and performed on 123 items, contains 43 figures, 27 tables, 5 appendices. The list of used sources contains 13 names.

Keywords: mobile app, pizza delivery, Swift, Firebase, sketch project, technical project, working project.

ЗМІСТ

ВСТУП

Доставка їжі - це завжди ціла подія й маленьке свято. Ми маємо можливість влаштувати вечерю зі стравами улюблених закладів й підтримати ресторанну сферу, не виходячи з дому. Ідея доставки народилася минулого сторіччя. Із часом ресторани, мережі, а згодом і окремі служби почали доставляти їжу на замовлення

Споживачі стали шукати способи заощадити час і спростити своє життя. Служби доставки продуктів у цьому сенсі стали справжнім порятунком, надаючи можливість замовляти продукти прямо зі свого будинку.

З розвитком технологій і появою мобільних застосунків служби доставки продуктів стали ще більш доступними і зручними. Це другий фактор. Клієнти можуть легко обирати продукти, стежити за статусом замовлення і оплачувати все онлайн. Технологічні інновації також дали змогу поліпшити системи відстеження, забезпечуючи більш точну і своєчасну доставку.

У відповідь на появу великої кількості служб доставки на ринку посилилася конкуренція. Це змусило компанії пропонувати вигідніші умови та різноманітніші акції, щоб залучити нових клієнтів, що стало третім фактором розвитку цих служб. Люди стали мати більше вибору, що стимулює служби доставки покращувати якість обслуговування і пропонованих послуг.

Однак найбільш значущим фактором у злеті популярності служб доставки продуктів стала пандемія коронавірусу. Захисні заходи, запроваджені для запобігання поширенню вірусу, включно з карантином і соціальним дистанціюванням, змусили людей обмежувати відвідування магазинів. У таких умовах служби доставки стали надійним і безпечним способом отримання необхідних товарів.

1 АНАЛІЗ ПОНЯТТЯ «МОБІЛЬНОГО ДОДАТКУ» ТА ПОСТАНОВКА ЗАДАЧІ НА РОЗРОБКУ ЗАСТОСУНКУ «PIZZA SHOP»

1.1 Опис предметної галузі

Мобільний додаток – це програмне забезпечення, яке встановлюють на мобільний пристрій (телефон або планшет). Найчастіше воно розробляється для різних операційних систем і може відрізнятися за структурою, функціональністю та призначенням.

Основна мета мобільного додатку – задовольнити потребу користувача. Потребою може бути будь-що: замовлення таксі, обробка фотографій або розгадування кроссфордів.

З моменту випуску першого мобільного телефону в 1983 році технології мобільних пристроїв продовжували еволюціонувати і призводити до розвитку нових функцій. Однією з найвагоміших та найцікавіших інновацій у цій галузі стали мобільні додатки. Сьогодні ми не уявляємо своє життя без них.

Історія появи першого мобільного додатка починається наприкінці 1990-х років, коли компанія Nokia випустила свій перший мобільний телефон з підтримкою Java ME. Ця технологія дозволяла створювати програми для мобільних пристроїв, але тоді вона була не дуже популярною, і мобільні програми не були особливо поширені.

Проте все змінилося у 2007 році, коли компанія Apple випустила свій перший iPhone. Цей телефон став справжньою революцією у світі мобільних пристроїв завдяки своєму сенсорному екрану та потужному процесору. Крім того, Apple випустила для нього свою власну операційну систему iOS, яка стала основою створення мобільних додатків.

Першим мобільним додатком для iPhone став додаток "Лотерея", випущений компанією Texas Lottery Commission у 2008 році. Він дозволяв грати в лотерею та перевіряти виграші прямо на телефоні. Ця програма була простою, але вона стала першим кроком у розвитку мобільних додатків.

Наступним кроком став випуск App Store у липні 2008 року, який став платформою для розповсюдження мобільних додатків для iPhone. За перший тиждень роботи App Store було завантажено понад 10 мільйонів програм, що стало явним підтвердженням того, що мобільні програми – це майбутнє.

З випуском iPhone 3G у 2008 році та iPhone OS 2.0 були додані нові функції для розробки мобільних додатків, такі як GPS-навігація та підтримка відео. Це відкрило нові можливості для розробників, і кількість мобільних програм на платформі iOS стала швидко зростати.

У 2010 році компанія Google відповіла на успіх App Store і випустила свій конкуруючий сервіс - Google Play (тоді він називався Android Market). Google Play пропонував більш відкритий підхід до розробки та розповсюдження програм, що залучило безліч розробників.

З тих пір кількість мобільних додатків на платформах iOS та Android почала зростати в геометричній прогресії, і в даний час мобільні додатки стали невід'ємною частиною нашого життя. Ми використовуємо їх для спілкування, роботи, розваг та багато іншого. Важливо, що мобільні додатки не обмежені лише платформами iOS та Android. Існує безліч інших операційних систем, таких як Windows Phone, BlackBerry OS, Tizen та інші, які також мають власні мобільні програми.

Таким чином, історія мобільних програм почалася з появи технології Java ME на мобільних телефонах Nokia і призвела до появи App Store і Google Play, де мільйони програм доступні для скачування та використання.[1]

1.2 Класифікація мобільних додатків

З розвитком мобільних додатків вони стали класифікуватися за різними особливостями їх роботи, такі як нативні додатки, веб-додатки, гібридні додатки. Нижче наведено кілька прикладів.

За особливостями роботи, мобільні програми прийнято ділити на 3 види:

- **Нативні додатки;**
- **Веб-додатки;**
- **Гібридні додатки.**

Нативні додатки

Нативні програми розробляються окремо для кожної платформи та дозволяють повністю використовувати всі можливості пристрою.

Для різних операційних систем розробники вибирають свою мову програмування. Для Android - Kotlin і Java, а для iOS - Objective-C або Swift. Приклад нативної програми – сервіс пошуку музики Shazam.

До речі, на сервіс музики Shazam можна націлюватися на програматик платформі BYYD. Наприклад, у разі він входить у добірку додатків під час просування автобренда.

Веб-додатки

Веб-додатки – це програми, які можна використовувати без завантаження та встановлення на смартфон, планшет або комп'ютер.

Девайс користувача в такому випадку отримує онлайн-доступ до даних без необхідності встановлення програми в постійну пам'ять. Приклад web-додатків: Інтернет-пошта (Gmail); текстові редактори (Google Документи).

Гібридні додатки

Гібридні програми – це суміш нативних та веб-додатків. Вони встановлюються як звичайні програми, але при цьому використовують WebView для завантаження веб-сторінок.

Це дозволяє їм мати доступ до даних пристрою та залишатися гнучкими, не вимагаючи встановлення окремого браузера. Як приклад – Uber, Evernote.

Однак мобільні додатки не обмежуються лише видами роботи. Ще одним важливим аспектом їхньої класифікації є поділ на різні типи.

Саме типи мобільних програм дозволяє зрозуміти для чого були створені програми.

1.3 Категорії мобільних додатків

У цій галузі виділяють чотири основні категорії:

- комерційні;
- соціальні;
- контентні;
- ігрові.

Комерційні програми

Комерційні додатки є додатками, у яких можна замовляти товари чи ж послуги і здійснювати оплату. Вони користуються популярністю серед компаній для залучення клієнтів та збільшення продажів.

Комерційні програми включають інтернет-магазини, де можна придбати одяг, взуття, косметику, а також програми для замовлення таксі або записи на консультацію до професіонала.

Соціальні платформи

Соціальні платформи, або іншими словами соціальні мережі, призначені переважно для спілкування через мобільні пристрої.

Крім того, на таких майданчиках можна спілкуватися, переглядати новини, а також продавати рекламу та просувати блоги, що корисно мати на увазі рекламодавцям.

Це сервіси, Telegram, Gmail тощо.

Контентні програми

Контентні програми відкривають користувачам швидкий доступ до актуальних даних. У них можна знайти інформацію про останні новини видань, дані про курси валют або корисні блоги на релевантні аудиторії тематики.

Таким чином, контентні платформи допомагають користувачам залишатися в курсі подій та розвиватися у різних галузях.

Прикладом контентної програми може бути програма для читання книг та журналів – Літрес.

Ігрові програми

Ігровими називають програми для мобільних пристроїв, таких як смартфони та планшети. Їхня головна мета – розважити користувача через інтерактивні ігри.

Ігрові програми можуть включати різноманітні жанри, від головоломок і аркад до стратегій і шутерів.[2]

1.4 «Pizza Shop» та його клони

У сучасному швидко змінюваному світі зручність доставки їжі додому – це більше, ніж просто розкіш, необхідність для багатьох. Додаток для доставки задовольняє цю потребу, пропонуючи плавний зв'язок між споживачами та різноманітним кулінарним вибором. Ця програма не тільки задовольняє потреби окремих людей, але й підтримує місцевий бізнес, розширюючи сферу його діяльності за межами традиційних клієнтів, які відвідують ресторани. Завдяки таким функціям, як індивідуальні рекомендації, рекламні пропозиції та зручний інтерфейс, найкращий додаток для доставки покращує якість обіду, роблячи його більш доступним та приємним.

Більше того, можливість відстежувати замовлення в режимі реального часу та отримувати оновлення безпосередньо на ваш мобільний пристрій підвищує рівень прозорості та довіри, гарантуючи, що ваша їжа буде доставлена так, як ви очікували. Саме такий рівень ефективності та зручності робить найкращий

додаток для доставки незамінним інструментом для сучасного життя, що забезпечує простий та надійний міст між кухнею та обіднім столом.

1) Uber Eats

Понад тисячу великих міст світу пропонують послуги UberEats. Через їхнє домінування на ринку багато ресторанів охоче співпрацюють з ними. У клієнтів буде широкий вибір варіантів, які сприятимуть позитивному зворотному зв'язку для зростання. UberEats – одна з найпопулярніших програм доставки їжі до США.

Незважаючи на те, що в Uber іноді трапляються помилки, і він не завжди найбільш зручний для користувачів для мільйонів людей, він все ще є кращим додатком для доставки їжі. У цьому полягає сила впізнаваності бренду.

Однак деякі функції програми виділяються, наприклад:

- Групові замовлення полегшують друзям та колегам поділити рахунок.
- Функція «Запланувати пізніше» дозволяє клієнтам відкласти доставку, доки вони не будуть готові їсти.
- Прості повторні замовлення дозволяють користувачам легко розміщувати замовлення на улюблені страви.

2) Just Eat

Just Eat – це європейський бренд онлайн-замовлень та доставки їжі. З ним співпрацюють понад 82 тисячі ресторанів, і він заробляє гроші на комісійних. Just Eat має 60 мільйонів щоденних користувачів.

Користувачі Just Eat мають унікальний вибір розміщувати замовлення на самовивіз. Крім того, бізнес дозволяє ресторанам використати свій кур'єрський персонал. Категорії страв у ньому різноманітні, зокрема вегетаріанські, безглютенові.

Чим відрізняється JustEat Takeaway?

- Простий та зручний інтерфейс.
- Відмінні можливості пошуку.
- Є багато купонів із кодами.

3) Doordash

DoorDash - одна з найстаріших та найвідоміших програм для замовлення їжі. Додаток працює більш ніж у 800 містах та має багато ресторанів-партнерів.

Клієнти можуть замовляти у національних або навіть місцевих ресторанах через Doordash. Функція планування доставки доступна у програмі для користувачів. За допомогою програми можна відстежувати доставку.

Що найкраще в DoorDash?

- Подача якісної їжі.
- Своєчасна доставка.
- Висока задоволеність клієнтів.

4) Deliveroo

Лондонський стартап Deliveroo пропонує послуги доставки їжі більш ніж у 200 містах. Найбільш поширений додаток для доставки їжі в Європі дозволяє користувачам замовляти їжу онлайн у відомих ресторанах та закусочних без типової їдальні. Компанія обслуговує понад 800 міст у 12 країнах.

У чому перевага Deliveroo?

- Більша різноманітність завдяки сумісності з нестандартними ресторанами.
- Доступ до знижок та купонів.

5) Foodpanda

FoodPanda – популярний онлайн-додаток для замовлення їжі, який доступний у 41 країні. Близько 40 тисяч найближчих ресторанів є партнерами, берлінський бізнес може доставляти їжу вчасно. Найбільшим ринком Foodpanda є Азія.

Гаманець, історія замовлень та кілька адрес – це лише деякі функції, які пропонує користувачам програма. Програма має систему відстеження замовлень, яка дозволяє користувачам відстежувати свою їжу.

Переваги FoodPanda:

- чисельні заклади харчування;
- знижки та бонуси, які легко знайти;
- легкі платежі.

6) Swiggy

Swiggy - популярний додаток для доставки їжі в Індії. Swiggy визнаний додатком №1 у Google Play з майже 1500000 завантажень. Майже кожне місто Індії має доступ до першого онлайн-додатку для доставки їжі, що співпрацює з 1040000 ресторанами.

Додаток досить простий у використанні. Вибираючи ресторан, він відображає оцінки, відгуки, вартість та інші важливі дані. Додаток також містить розділи із закусками, основними стравами та десертами.

У чому найкращий Swiggy?

- Безперебійний GPS для допомоги користувачам із замовленням у ресторані у цьому районі.
- Він орієнтований на ринок Індії.
- Функція гаманця дозволяє користувачам заощаджувати гроші та використовувати їх для оплати покупок.

7) Zomato

Zomato – це індійська компанія з доставки їжі, що працює у 25 країнах. Якщо хтось хоче поїсти поза домом, додаток пропонує доставку їжі та рекомендує найкращі ресторани.

Програма добре відома тим, що використовує соціальні мережі реклами. Маркетингова стратегія Zomato та високоякісні послуги допомогли їй завоювати шанувальників серед індійських клієнтів. Бізнес-модель Zomato унікальна та ефективно приваблює користувачів.

Чому людям подобається Zomato?

- Простий обмін інформацією майже як соціальні мережі для гурманів.
- Комплексні зручні інструменти пошуку.
- Просте сортування відповідно до уподобань користувача.
- Інтуїтивно зрозуміла база знань для вирішення проблем клієнтів.
- Спрощені функції оглядів.

8) Пицца Domino's

Domino's – популярний застосунок доставки піци із першокласною программою

Ця програма, на відміну від інших програм у цьому списку, доставляє їжу лише з ресторану.

Переваги додатку:

- Чітко відображаються зображення доступних замовлень для користувачів.
- Можливість змінювати начинку, змінювати замовлення та повністю персоналізувати страву.
- Відстеження замовлень відображає кожний етап процедури, включаючи підготовку, випікання, отримання замовлення та доставку.

9) Talabat

Кувейтська компанія на замовлення їжі Talabat працює у семи країнах Близького Сходу. Понад 2 мільйони людей активно використовують додаток, який пропонує понад 10 000 ресторанів.

Програма має зрозумілий та нехитрий інтерфейс. На момент доставки замовлений товар можна відстежувати. Програма також підтримує онлайн-платежі.

10) Glovo

Glovo прискорило своє зростання після запуску у Барселоні у 2015 році. Зараз Glovo працює у 200 містах у 26 країнах, забезпечуючи якнайшвидшу доставку їжі.

До програми підключено понад 20 тисяч ресторанів. Число активних користувачів понад 10 мільйонів.

Чому людям подобається Glovo?

• Простий у використанні інтерфейс.

• Високі можливості пошуку.

• Швидка доставка.

• Високий рівень задоволеності клієнтів.[3]

1.5 Основні компоненти мобільного додатку

Важливим компонентом розробки мобільного додатку є Frontend та Backend. Це дві невід'ємні частини розробки мобільного додатку. Хоча іноді замовчують про важливість роботи кожного розробника в команді.

Frontend – це розробка інтерфейсу користувача і функцій, які працюють на стороні клієнта певним чином. Іншими словами – це все, що додаток може читати і виводити перед користувачем на екран або запускати. Фронтенд являє собою публічну сторону додатку, з якою людина може взаємодіяти, встановлюючи контакт напряму.

Розробка фронтенда передбачає кропітку роботу, в результаті якої кожна ікона, кнопка або текст стоять на своєму місці, виглядають цілісно, не заважають і не перекривають один одного. При цьому значення має не тільки зовнішній дизайн, та ще щоб всі його елементи виконували своє пряме призначення, тобто з їх допомогою можна було здійснити необхідні дії.

Тому для випуску якісного продукту фронтенд-розробнику доведеться налагодити взаємодію з іншими фахівцями: програмістами, маркетологами, дизайнерами та іншими.

Він включає в себе наступні ключові елементи:

- **Користувацький інтерфейс (UI)**: Візуальна частина додатка, що включає екрани, кнопки, форми, анімації та інші елементи інтерфейсу. Якісний UI забезпечує привабливість додатка та інтуїтивну взаємодію користувача з ним.
- **Користувацький досвід (UX)**: UX охоплює логіку та дизайн взаємодії користувача з додатком, забезпечуючи зручність та інтуїтивність використання. Добре продуманий UX підвищує задоволеність користувачів і збільшує їх залученість.
- **Клієнтська логіка**: Код, що виконується на пристрої користувача, який обробляє введення даних, відображає інформацію та взаємодіє з сервером. Клієнтська логіка відповідає за виконання локальних обчислень та представлення даних.

Бекенд – це все, що працює на сервері. Виходячи з цього, бекенд розробка – це робота над програмними засобами, спрямованими на реалізацію логіки ресурсу. Ця частина прихована від очей користувача, оскільки відбувається за межами його додатку.

Розробник додатку в даному випадку використовує ті ресурси, які є на сервері. При цьому його обов’язки можуть значно варіюватися, залежно від того про який продукт йде мова. Так, фахівець може займатися створенням, інтеграцією баз даних, забезпечувати безпеку ресурсу, налаштовувати технології резервного копіювання або ж відновлення інформації.

Він включає в себе наступні ключові елементи:

- **Серверна логіка**: Логіка обробки даних і виконання операцій на сервері, включаючи управління користувачами, обробку замовлень, розрахунок вартості та інші важливі функції. Серверна логіка забезпечує надійну та масштабовану роботу додатка.
- **База даних**: Сховище даних додатка, що включає інформацію про користувачів, замовлення, продукти та інші сутності. Ефективне управління базою даних важливе для швидкого доступу та надійного зберігання інформації.

- **API (Application Programming Interface)**: Інтерфейси для взаємодії між фронтендом і бекендом, а також для інтеграції із зовнішніми сервісами, такими як платіжні системи та служби геолокації. API забезпечує стандартизований обмін даними та інтеграцію з різними системами [4,5].

1.6 Опис проєкту

Проект, що розробляється – «Pizza Shop» – це варіація багатьох додатків для замовлення. Її особливість полягає в тому, що додаток простий та інтуїтивно зрозумілий, а також в додатку одночасно можливо відстежувати етапи виконання замовлення.

Ціль додатку – залучити максимальну кількість користувачів та надати можливість швидко замовити їжу, без попередніх дзвінків с поясненням, що вам потрібно.

Додаток включає в себе: продукцію, обліковий запис, інформацію користувача та базу даних.

Продукція – це колекція елементів, в яку можливо додавати нову продукцію та обирати її.

Обліковий запис – це сутність користувача, яка дозволяє виконувати вхід в додаток та дає йому унікальний ключ в базі даних для подальшого використання додатку. Також при бажані залишатися у додатку при закриті.

Інформація про користувача – це дані користувача, які по унікальному ключу зберігаються у базі даних його: ім’я, номер телефону, адреса та його замовлення.

База даних додатку - це організована структура, яка призначена для зберігання, зміни та обробки взаємозалежної інформації.

Які структури даних включає в себе:

- Замовлення(Orders).

- Продукції(Products).
- Користувачів(Users).

Замовлення(Orders) – це структура даних, яка включає в себе інформацію про поточні замовлення: користувачів які замовили, час коли було замовлено та етапи виконання замовлення.

Продукції(Products) – це структура даних, яка містить у собі інформацію про поточні товари: опис, назву, ціну. Саме сюди додаються нові товари.

Користувач(User) – це структура даних, яка містить інформацію про користувача: адресу, ім’я, номер телефону та їх унікальний ключ. Сюди додаються нові користувачі [5].

На рисунках 1.1 – 1.3 відображено вище описані структури даних у базі даних.

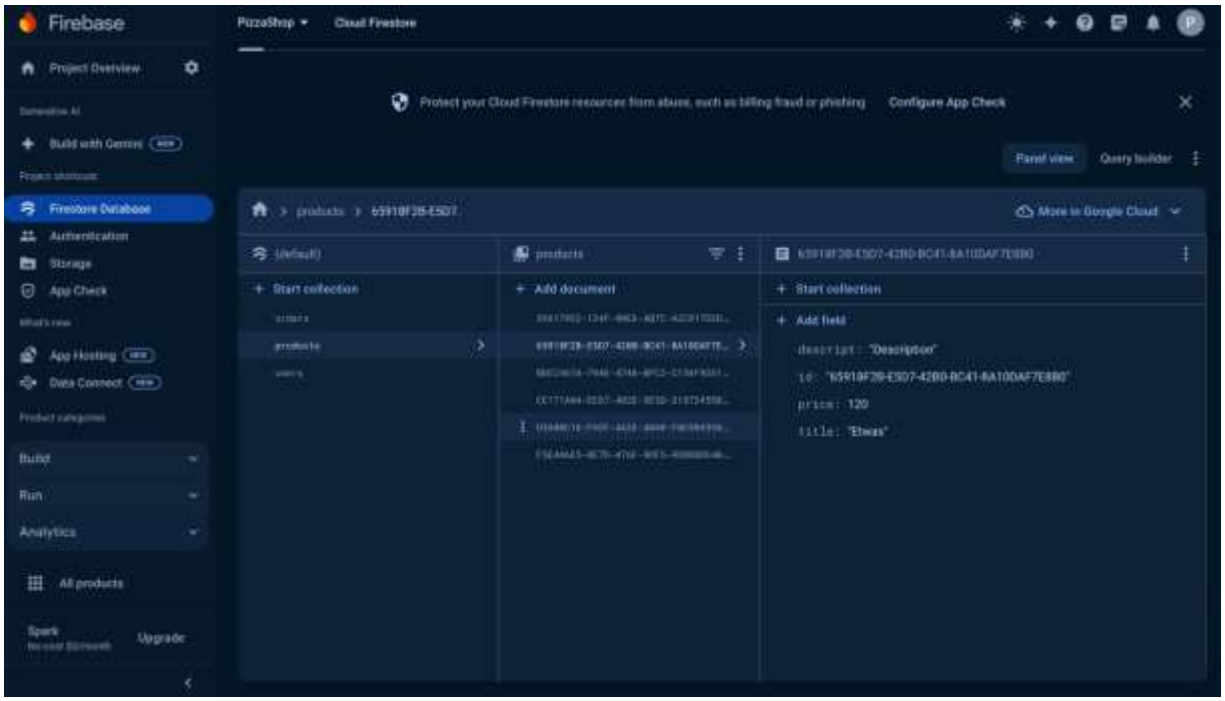

Рисунок 1.1 – Структура даних Продукти

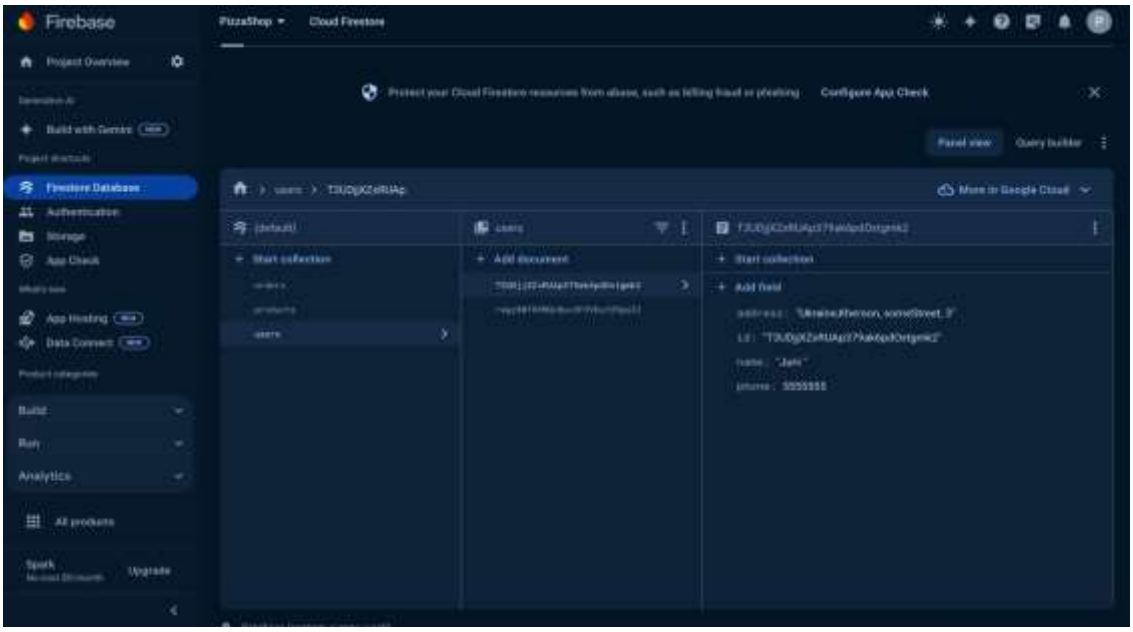

Рисунок 1.2 – Структури даних Користувачі

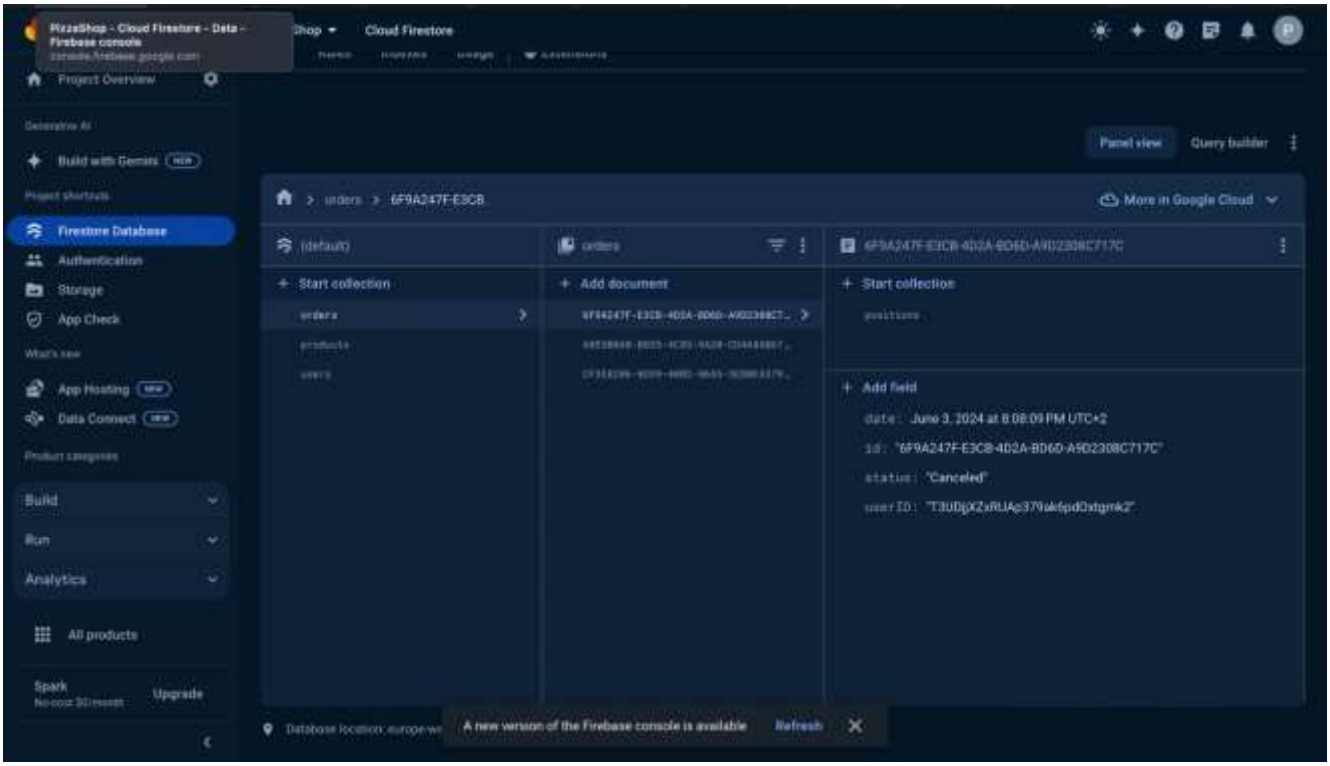

Рисунок 1.3 – Структури даних Замовлення

1.7 Основні вимоги до проекту.

Основні функціональні вимоги

Додаток повинен бути багатокористувацьким, а це значить, що крім самого додатку також повинна бути база даних.

База даних повинна надавати можливість адміністратору створити позицію/продукцію або надати можливість користувачу створити обліковий запис, зберегти його та зробити деякі налаштування зі сторони користувача у структурі даних.

Після створення продукції, база даних повинна відправляти її у додаток для відображення нової продукції у колекції, а саме:

- назву продукції,
- фото продукції,
- ціну,
- опис продукції,

Також при створені облікового запису зберігати:

- пошту користувача,
- пароль,

дані користувача, такі як:

- номер телефону,
- адрессу
- ім’я
- поточні замовлення від користувача.

При вході в обліковий запис користувачу повинні бути доступні такі опції:

- Можливість додати або змінити дані користувача.
- Здійснити замовлення.
- Вийти з додатку при бажанні.

При запуску додатку, відображається екран авторизації, якщо не було входу попередньо, у цьому екрані має відобразитися поля введення, в які він повинен ввести пошту, а також пароль, якщо облікового запису не існує то створити новий, при цьому увести пошту, пароль та підтвердити пароль.

При вході у додаток, користувачеві має відобразитися «Каталог» з наступними можливостями:

- Продивитися товар.
- Вибрати товар.
- Перейти до профілю.
- Перейти до корзини.

Якщо вибрати «Корзину», користувач може подивитися свій вибір та підтвердити його чи відмінити.

Якщо вибрати «Профіль», користувач може заповнити дані про себе, подивитися його поточні замовлення та при необхідності вийти з профілю.

При виборі «Товару», користувач може вибрати розмір та побачити с чого складається піца.

Нефункціональні вимоги:

- Додаток буде розроблений для операційної системи IOS
- Додаток повинен бути легким для памяті (не займати багато простору у внутрішній пам’яті)
- Додаток повинен працювати Онлайн - без крахів, щоб у користувача був добрий досвід користування
- Додаток повинен мати простий та інтуїтивно зрозумілий інтерфейс
- Додаток повинен бути зрозумілий всім користувачам, а тому основною мовою має бути англійська

1.8 Постановка задачі

З викладених вимог до програмного продукту була розроблена постанова задачі.

Головною метою даної роботи є розробка програмного забезпечення, яке відповідає наступним вимогам:

1) Вимоги до складу виконуваних функцій

а) «Авторизація»:

- Створення облікового запису:

Вхідна інформація: введення пошти, паролю, та підтвердження паролю.

Вихідна інформація: завантаження даних входу облікового запису до бази.

- Перевірка наявності облікового запису:

Вхідна інформація: пошта користувача та пароль.

Вихідна інформація: перевірка пошти та паролю.

- Вхід до облікового запису:

Вхідна інформація: Відсутня.

Вихідна інформація: завантаження даних користувача.

б) «Колекція»:

- Відображення колекції:

Вхідна інформація: код відображення колекцій.

Вихідна інформація: отримання даних колекції.

- Натискання на колекцію:

Вхідна інформація: код колекції.

Вихідна інформація: переміщення до продукції.

- Переміщення до продукції:

Вхідна інформація: отримання опису з бази.

Вихідна інформація: Опис продукції.

- Вибір розміру та кількості продукції:

Вхідна інформація: поточна сума продукції.

Вихідна інформація: загальна сума.

- Додавання до кошика:

Вхідна інформація: загальна сума, кількість позицій.

Вихідна інформація: відправлення до кошика.

в) «Кошик»:

- Відображення поточних замовлень:

Вхідна інформація: Відсутня.

Вихідна інформація: відображення вибраної продукції та ціни.

- Замовлення піци:

Вхідна інформація: Загальна сума.

Вихідна інформація: Відправлення замовлення до адміністратора та відображення вибраного замовлення у профілі користувача.

г) «Обліковий запис»

- Відображення профілю користувача:

Вхідна інформація: отримання даних користувача.

Вихідна інформація: перегляд поточного замовлення та стадій його виконання.

- Вихід з облікового запису:

Вхідна інформація: поточний обліковий запис.

Вихідна інформація: вихід до «Авторизації».

- Обліковий запис Адміністратора:

Вхідна інформація: пошта, пароль.

Вихідна інформація: перевірка даних входу у базі.

- Відображення поточних замовлень:

Вхідна інформація: отримання даних.

Вихідна інформація: дані про користувача, його замовлення та стадії замовлення.

- Перевірка позицій замовлення:

Вхідна інформація: отримання з бази.

Вихідна інформація: відсутня.

- Додавання нових продукцій:

Вхідна інформація: назва, опис, ціна.

Вихідна інформація: відправка до бази.

2) Вимоги до вхідної і вихідної інформації

Вхідною інформацією для даної системи можуть бути як події натискання кнопок, переміщень до відображення, так і дані користувача та продукції, котрі передаються.

Вихідною інформацією для даної системи можуть бути як завантаження даних до бази даних або із неї, так і різні дії користувача.

3) Вимоги до інформаційної і програмної сумісності

Для запуску додатку на телефоні повинна бути встановлена операційна система - IOS 10.3 та вище.

4) Вимоги до складу й параметрів технічних засобів

Для запуску додатку, потрібен мобільний телефон зі системними характеристиками, котрі не будуть нижчі за наступні:

- процесор AppleA6 з тактовою частотою 1,3 GHz та двома ядрами;
- оперативна пам'ять з об'ємом 1GB;
- з об'ємом пам'яті 1GB;
- не менше 15MB вільного дискового простору.

Детальніше постановка задачі сформована в технічному завданні, що наведене у Додатку А.

2 РОЗРОБКА ПРОЕКТУ ДОДАТКУ «Pizza Shop»

При розробці проекту програмного забезпечення були виконані роботи з ескізного, технічного та робочого проекту. Для цього використовувалися об'єктно-орієнтовані методології та засоби UML. Результатом роботи є розроблений додаток «Pizza Shop»

2.1 Ескізний проект

На етапі розробки ескізного проекту будуть представлені наступні елементи: діаграма варіантів використання, специфікація варіантів використання, діаграми діяльності, концептуальна модель, а також буде представлений проект користувацького інтерфейсу.

2.1.1 Розробка моделі варіантів використання

Модель варіантів використання – це модель планованих функцій системи та навколосистемних об'єктів, яка служить як угода між замовником та розробниками. Варіанти використання є сполучним елементом, який проходить через всі етапи розробки системи.

Розроблюваний додаток має надавати можливість виконувати дії, як зі сторони клієнта так зі сторони адміністратора. Для реалізації цього мають бути дві системи. Перша система повинна відповідати за зчитування дій користувача та відображення інтерфейсу для нього. Інша система має бути системою для адміністратора. Отже, для візуалізації було побудовано модель варіантів використання [6].

Для побудови моделі варіантів використання були використані:

- Елемент «актор» у кількості трьох штук: User, Admin, Base.

- Елемент «система» у кількості двох штук: User System, Admin System.
- Елемент «прецедент» у кількості дванадцяти штук: Create Profile, Enter Profile(User System), Receive Data(User System), Order Selection,Confirm View, User Information, Send Data(User System), Leave Profile(User System), Enter Profile(Admin System), Receive Data(Admin System), Add Products, Send Data(Admin System), Leave Profile(Admin System).

На рисунку 2.1 зображено модель варіантів використання для додатку «Pizza Shop». На діаграмі відображені основні відносини між Акторами та системами, що відбуваються завдяки прецедентам.

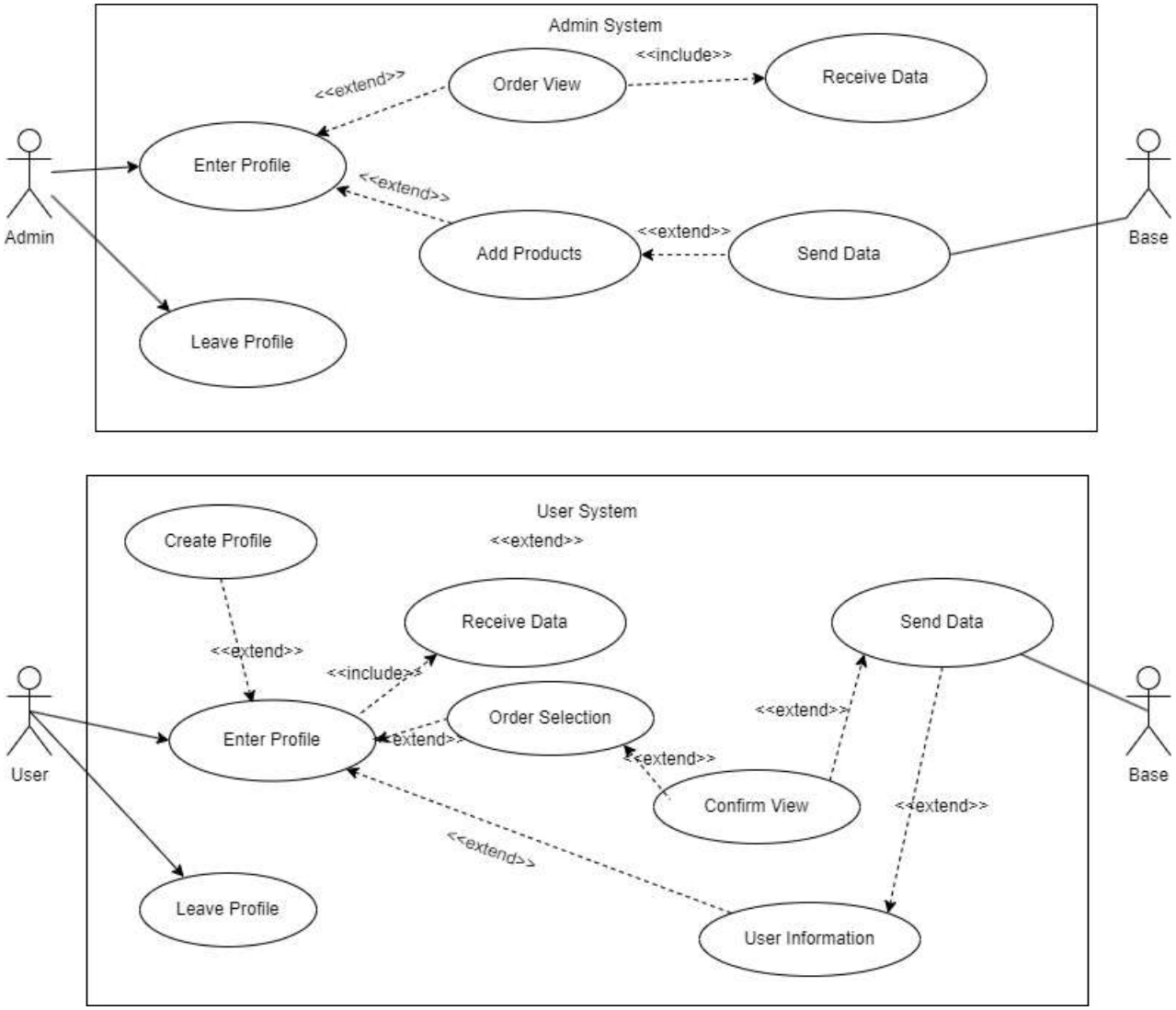

Рисунок 2.1 – Діаграма варіантів використання додатку «Pizza Shop»

2.1.2 Специфікації варіантів використання

Для кращого розуміння діаграми варіантів використання, яка наведена раніше, у цьому підрозділі будуть описані специфікації.

Опис акторів

User – головна дійова особа додатку, яка виконує певні функції для досягнення певних цілей в додатку

Admin – особа, яка може переглядати поточні замовлення від користувачів та створювати нові замовлення для них.

Опис систем.

User System – це середовище інтерфейсу та бізнес-логіки для користувача, за допомогою яких він досягає цілей додатку.

Admin System – це середовище інтерфейсу та бізнес-логіки для Адміністратора, які надають йому змогу дивитися за всіма замовленнями від користувачів та обміну даними із сервером.

Специфікації

Create Profile:

- Головний актор: User.
- Опис: дана дія служить для того, щоб користувач ввів пошту та пароль для створення облікового запису, користувачу призначується унікальний ключ.
- Передумова: якщо пошта та пароль введені не вірно то обліковий запис не створюється.
- Основний потік:
  - користувач вводить початкові дані,
  - користувач натискає кнопку створення облікового запису.
- Пост умова: створюється обліковий запис, та записуються до бази.
- Альтернативний потік: відсутній.

Enter Profile

- Головний актор: User та/або Admin.
- Інші учасники: відсутні.
- Опис: дана дія служить для входу в обліковий запис, User входить на початковий екран вибору продукції. Якщо вхід виконується актором Admin, відображається інформація про замовлення.
- Передумова:
  - Вхід виконаний,
  - у разі входу актором User, актор входить на початковий екран додатку,
  - у разі використання дії актором Admin, актор входить на екран Відображення поточних замовлень .
- Основний потік:
  - у разі використання входу актором User або Admin, система перевіряє унікальний ключ облікового запису.
- Альтернативний потік: відсутній.

Receive Data

- Головний актор: User System або Admin System у якості системи
- Інші учасники: User, Admin
- Опис: дана дія слугує для прийняття даних, котрі на даний момент знаходяться у базі та оновлюються у разі змінення.
- Передумова:
  - додаток запущений, та отримує дані
- Основний потік:
  - User виконує якісь дії,
  - на основі виконаних дій генерує дані,
  - дані надсилаються до Адміністратора,
  - дані оновлюються в базі.

- Альтернативний потік: відсутній

Leave Profile

- Головний актор: User, Admin.
- Інші учасники: відсутні.
- Опис: дана дія слугує для виходу із облікового запису, для User та Admin. У користувача вихід знаходиться у Профілі, у Адміністратора у вікні відображення замовлень
- Передумова: певна функція вибрана.
- Основний потік:
  - Профіль користувача відображає варіанти виходу,
  - Адміністратор виходить,
- Альтернативний потік: відсутній.

Send Data

- Головний актор: User System, Admin System.
- Інші учасники: Admin, User
- Опис: дана дія слугує для відправки всіх даних до бази, котрі були зміненні, даних з обробленою інформацією.
- Передумова: дані оброблені та готові для відправки до бази.
- Основний потік: дані відправляються до бази.
- Пост умова: дані приймаються та оновлюються.
- Альтернативний потік: відсутній.

User Information

- Головний актор: User.
- Інші учасники: User System у якості системи.
- Опис: дана дія слугує для введення даних користувача та відображення його замовлень, котрі відправляться після їх підтвердження.
- Передумова: текстові поля порожні та очікують даних користувача.

- Основний потік:
  - Текстові поля очікують вводу
  - дані зберігаються,
  - Відправляються до бази.
- Альтернативний потік:
  - дані не зберіглись,
  - стан Профілю не оновився.

Add Products

- Головний актор: Admin.
- Інші учасники: Admin System у якості системи.
- Опис: дана дія слугує для додавання нових позицій.

Передумова:

- Функція обрана,
- Відображається екран додавання,
- Всі дані позиції введено.

Основний потік:

- формується позиція,
- Позиція відправляється до бази.

Альтернативний потік: у разі не ведення всіх даних позиції, до бази нічого не відправляється.

Order Selection

- Головний актор: User
- Інші учасники: відсутні.
- Опис: дана дія слугує для вибору продукції користувачем, її розміру та кількості.
- Передумова: користувач натискає на продукцію, відкривається екран вибору кількості та розміру.
- Основний потік:

- користувач обирає продукцію,
- по натисканню відкривається вікно вибору.
- після вибору користувач натискає кнопку відправки до корзини

• Пост умова: після вибору та натиску обрана продукція відправляється до (Confirm View)..

• Альтернативний потік: якщо користувач хоче змінити замовлення він повертається на попередній екран.

Confirm View

• Головний актор: User.

• Інші учасники: User System у якості системи.

• Опис: дана дія слугує для підтвердження замовлення, відображенню ціни за замовлення.

• Передумова: продукція обрана.

• Основний потік: користувач підтверджує замовлення.

• Пост умова: Замовлення відправляється до бази та Адміністратору.

• Альтернативний потік: Можливо відмінити замовлення.

Order View

• Головний актор: Admin.

• Інші учасники: Admin System.

• Опис: дана дія слугує для відображення замовлень від користувача

• Передумова: Адміністратор отримав дані замовлень.

• Основний потік:

- Адміністратор отримує дані від користувача,
- Відображається час замовлення, позиції та стан її виконання,

• Пост умова: отримана Дані від бази.

• Альтернативний потік: відсутній.

2.1.3 Сценарії основних варіантів використання

Діаграма діяльності – це графічне представлення робочих процесів, алгоритмів із підтримкою вибору, ітерації та паралельності. Вона дозволяє моделювати послідовність дій, потік управління, паралельні процеси.[7]

Для детальнішого розуміння специфікацій у цьому розділі будуть представлені діаграми діяльності основних варіантів використання (рисунки 2.2 – 2.13).

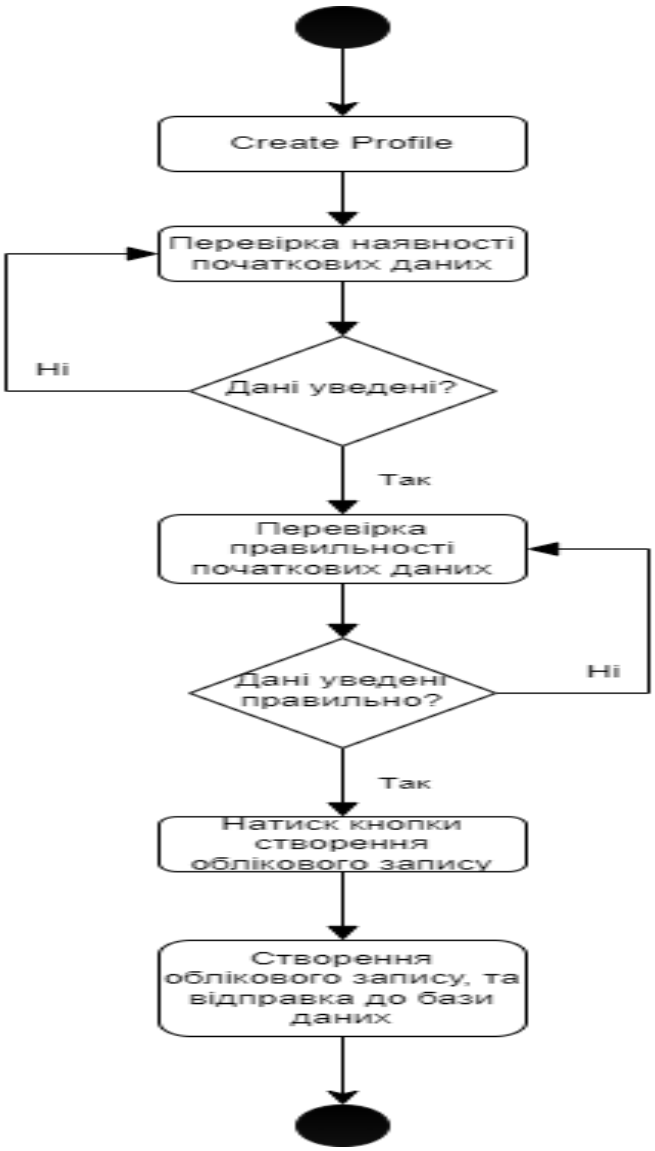

Рисунок 2.2 – Діаграма діяльності варіанту використання для створення облікового запису.

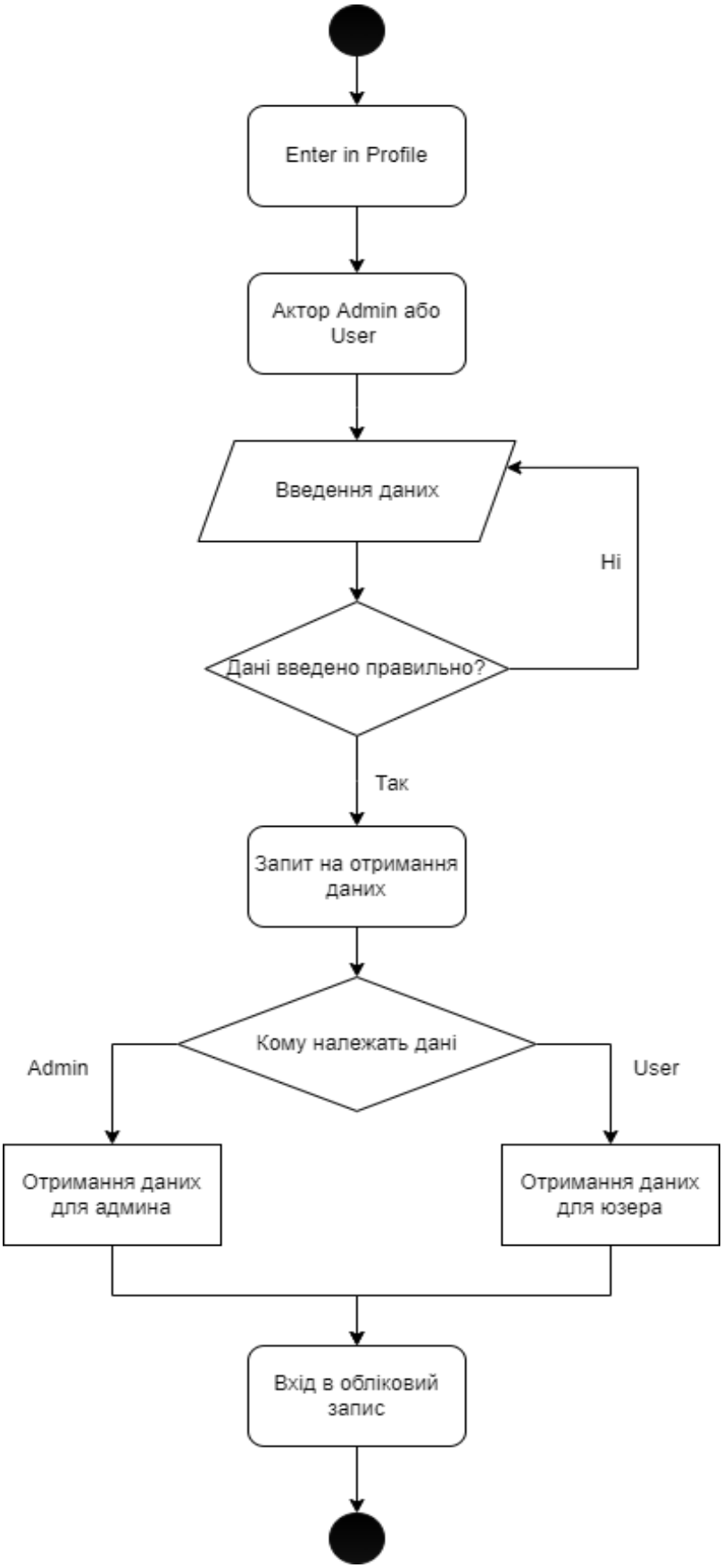

Рисунок 2.3 – Діаграма діяльності варіантів входу до облікового запису.

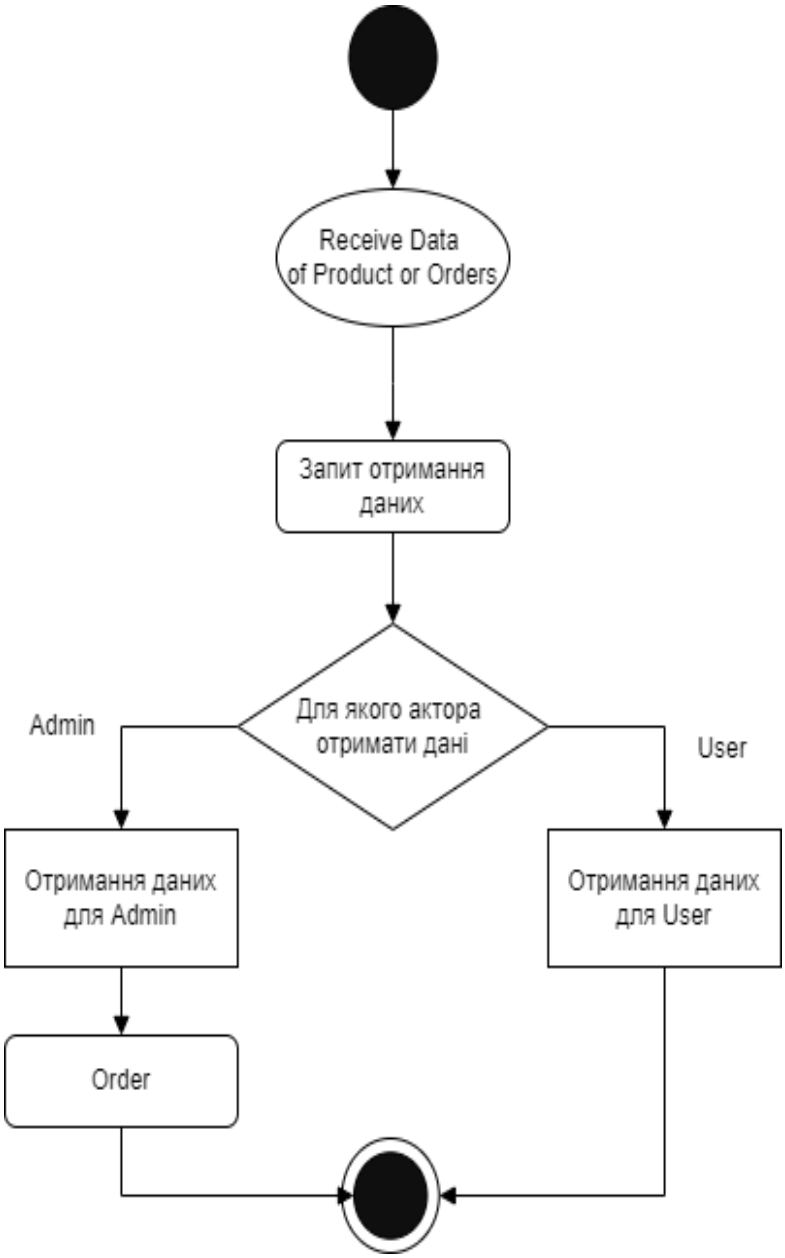

Рисунок 2.4 – Діаграма діяльності варіантів отримання даних користувачем або адміністратором.

.

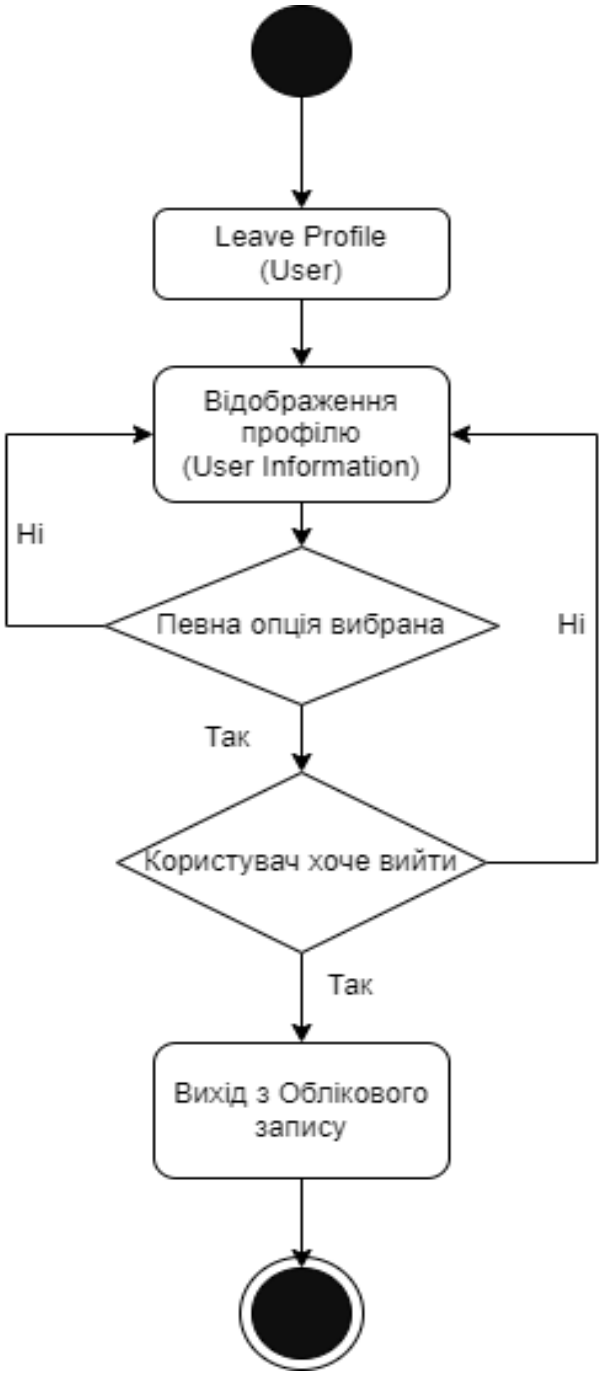

Рисунок 2.5 – Діаграма діяльності варіанту використання виходу з облікового запису Користувача.

Рисунок 2.6 – Діаграма діяльності варіанту використання для вибору замовлення.

Рисунок 2.7 – Діаграма діяльності варіанту використання для Підтвердження замовлення та відправлення до бази.

Рисунок 2.8 – Діаграма діяльності варіанту використання для відправлення даних до бази

Рисунок 2.9 – Діаграма діяльності варіанту використання для відправлення даних

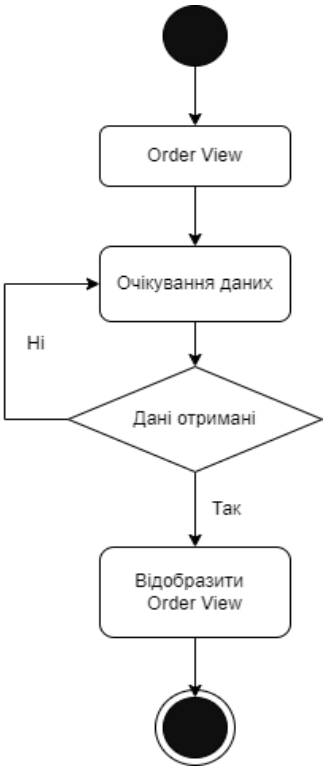

Рисунок 2.10 – Діаграма діяльності варіанту використання для відображення замовлень

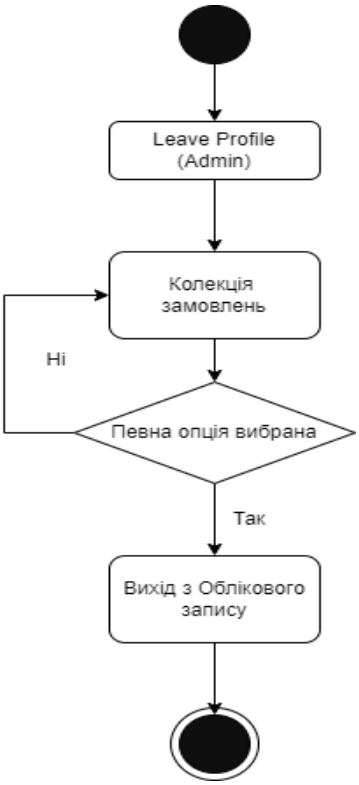

Рисунок 2.11 – Діаграма діяльності варіанту використання виходу з облікового запису Адміністратора

Рисунок 2.12 – Діаграма діяльності варіанту використання для додавання нової позиції

2.1.4 Інформаційна модель

Концептуальна модель – це абстрактна модель предметної області, що складається зі списку понять, які використовуються для опису предметної області, а також властивостей та характеристик, класифікації понять [8].

Нижче наведена концептуальна модель, яка представлена у вигляді діаграми «сутність-зв'язок» (рисунок 2.14). На діаграмі відображено основну частину даних для розуміння структури проекту та взаємодії його елементів між собою.

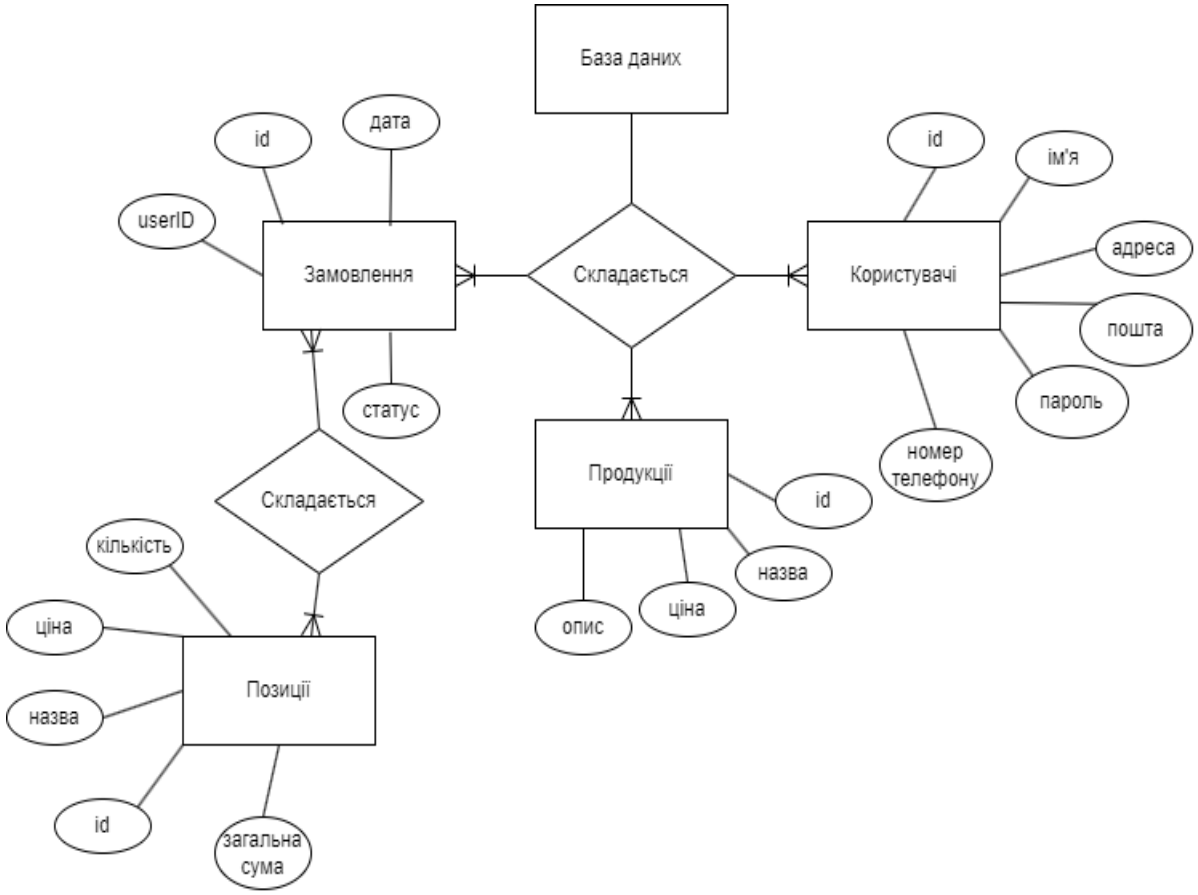

Рисунок 2.13 – Концептуальна модель даних для системи

2.1.5 Проект користувальницького інтерфейсу

Інтерфейс користувача або UI – це засіб взаємодії людини з програмним забезпеченням або апаратним оснащенням. Інтерфейс користувача представляє набір різних елементів, які розташовані у правильній послідовності, що надає користувачеві зручне його використання. Найпоширенішими є графічний, текстовий та інтерфейс командного рядка, які використовуються в тій чи іншій системі залежно від ситуації [4,6].

У рамках розроблюваного проекту використовується графічний інтерфейс, який містить мова програмування Swift – це SwiftUI.

На рисунках 2.14 – 2.16 представлені основні інтерфейси, які використовують Адміністратор та користувач.

Рисунок 2.14 – Інтерфейс стартового вікна Авторизації

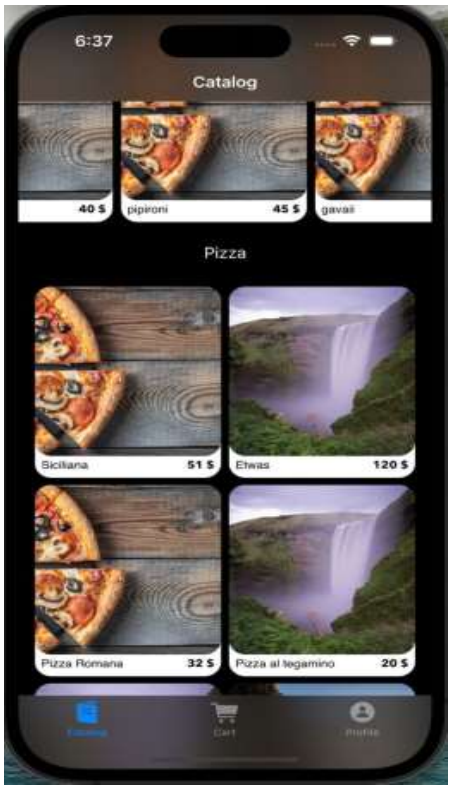

Рисунок 2.15 – Інтерфейс екрану Каталог

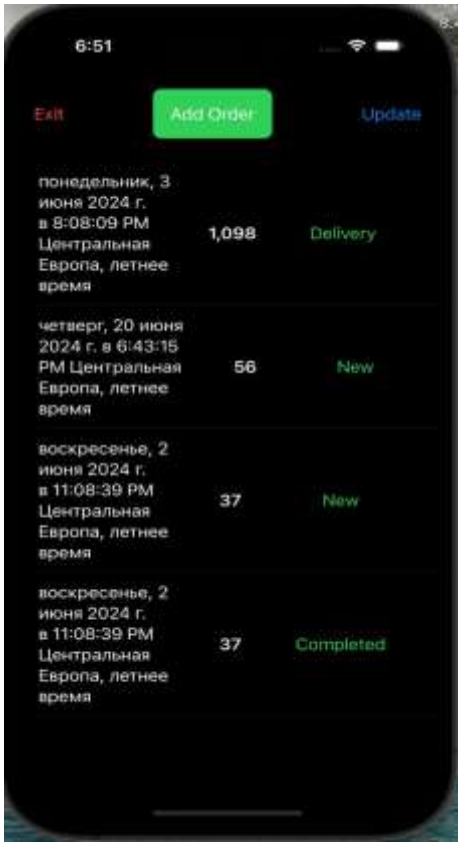

Рисунок 2.16 – Інтерфейс Адміністратора «AdminOrderView»

2.2 Технічний проект

Технічний проект – стадія розробки системи, а також документ, основна мета якого – представлення остаточних проектних рішень з розробки програмного забезпечення.

У рамках технічного проекту програмного забезпечення було створено статичну модель програми (діаграма класів), описані специфікації класів, діаграми послідовності та логічна модель даних.

2.2.1 Статична модель програми

Статична модель – це важливий елемент проектування системи, який дозволяє уявити структуру та компоненти програмного продукту. Досить часто ця модель використовується для візуалізації архітектури програмного забезпечення, що дозволяє краще зрозуміти зв'язок між компонентами.

Для представлення даної моделі буде використана діаграма класів (рисунки 2.17 та 2.18).

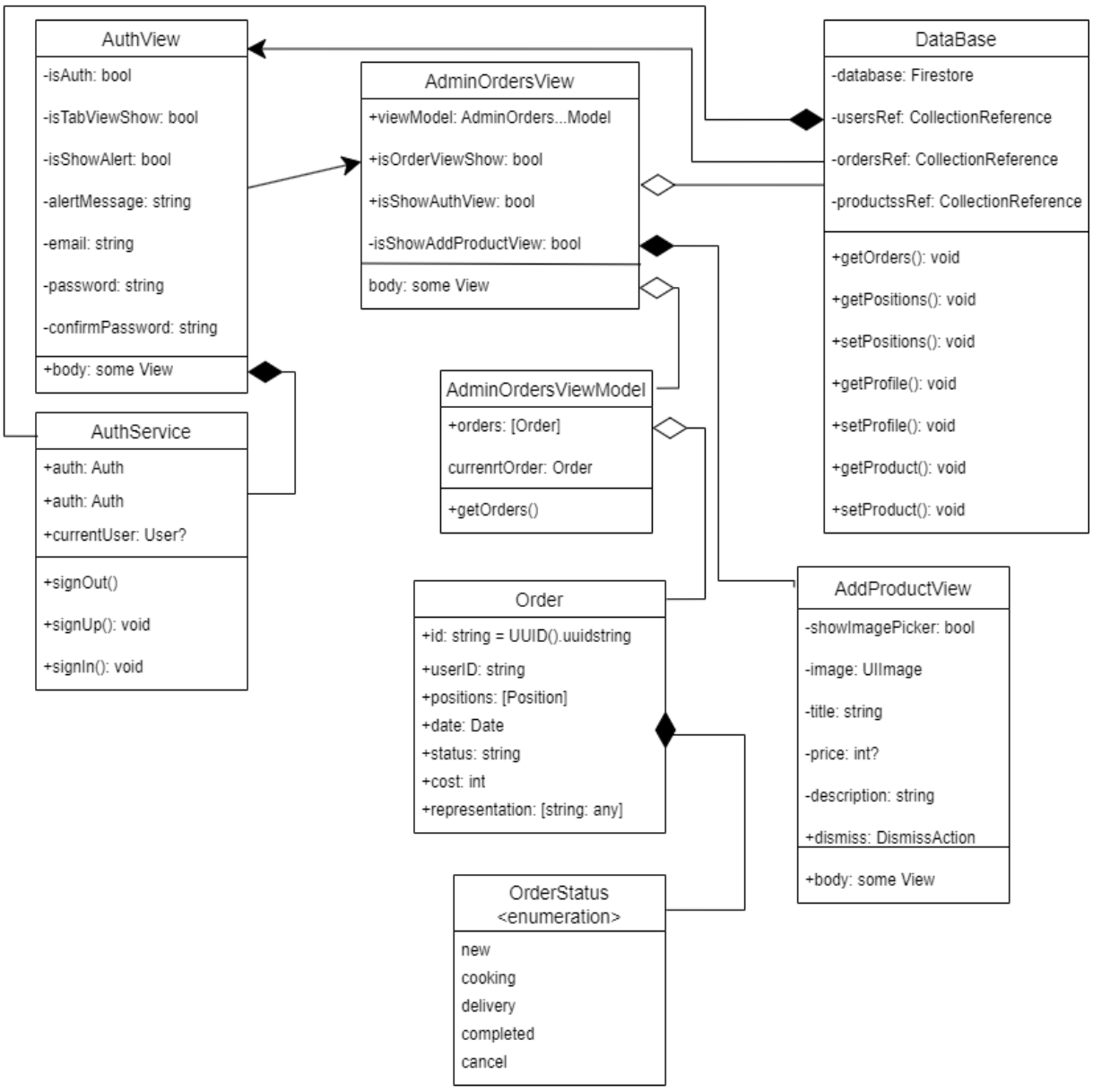

Рисунок 2.17 – Діаграма класів системи Admin System

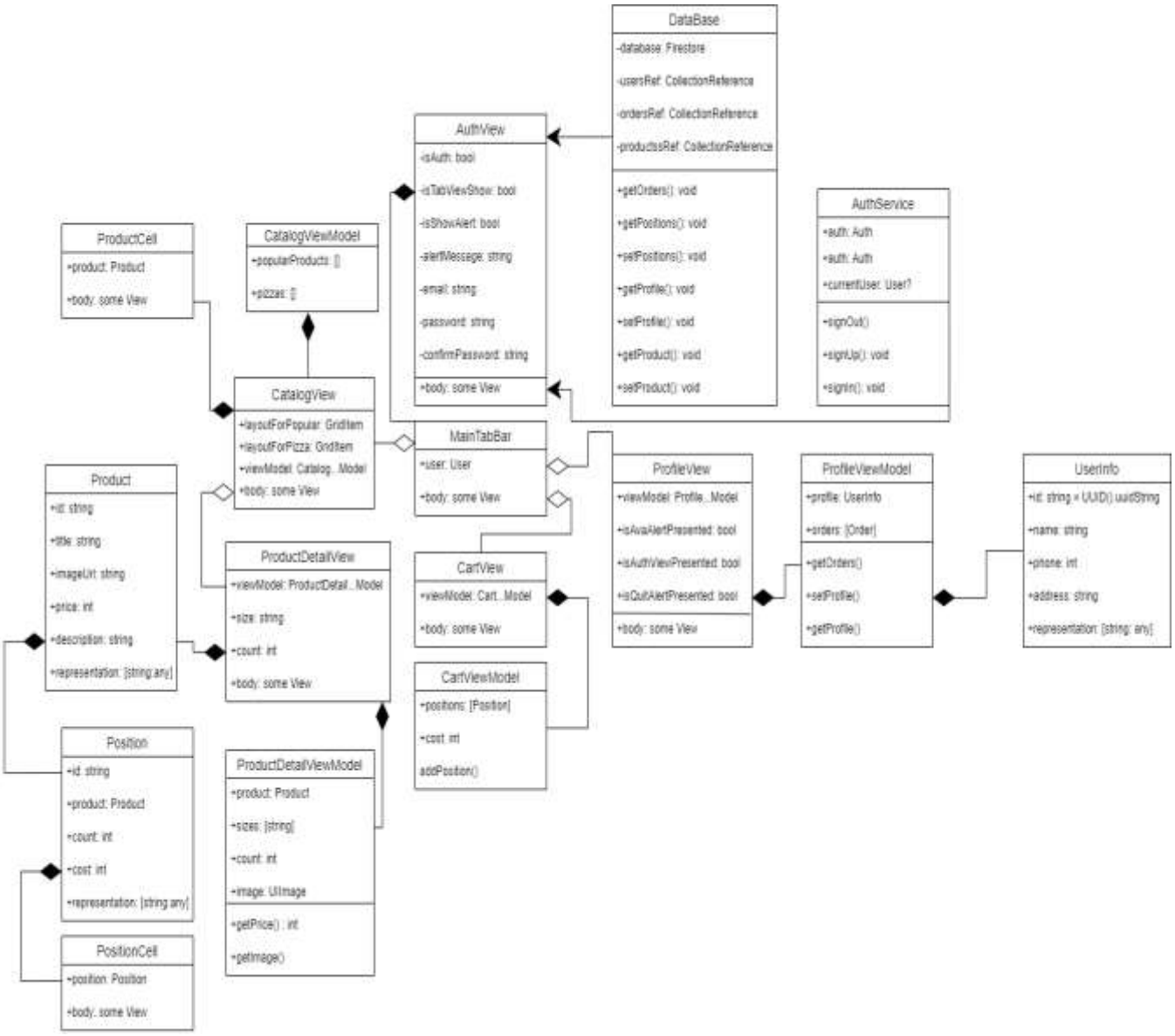

Рисунок 2.18 – Діаграма класів системи User System

2.2.2 Специфікації класів

Кожен із класів, котрі наведені вище, має своє призначення. Кожен атрибут використовується у логіці системи, а кожен метод виконує частину логіки. Нижче наведені описи класів.

Система Admin та User:

Клас AuthView.

- Опис: головний клас програми – точка входу. Виконує відкриття екрану користувача або адміністратора після авторизації.
- Атрибути: isAuth, email, password, confirmPassword, isTabViewShow, isShowAlert, alertMesssage, body(UI).
- Методи: відсутні

Клас AuthService.

- Опис: відповідає за вхід та вихід або створення облікового запису.
- Атрибути: currentUser, auth.
- Методи:

  signOut() – виконує функцію виходу з облікового запису,

  signIn() – виконує функцію входу в обліковий запис,

  signUp() – виконує функцію створення облікового запису.

Клас DataBase

- Опис: отримує, відправляє або зберігає усю інформацію про: користувача, замовлення, продукцію.
- Атрибути: db, usersRef, ordersRef, productsRef.
- Методи:

getOrders() - отримує замовлення з бази

setOrders() - відправляє замовлення до бази

getPositions() - отримує позиції з бази

setPositions() – додає позиції до бази

setProfile() – додає обліковий запис до бази

getProfile() – отримує обліковий запис з бази

setProduct() – додає продукцію до бази

getProducts() – отримує продукцію з бази

Система Admin:

Клас AdminOrdersView

- Опис: відображає поточні замовлення користувачів також кнопки: додати замовлення та вихід з профілю.

- Атрибути: viewModel, isOrderViewShow, isShowAuthView, isShowAddProductView, body.
- Методи: відсутні.

Клас AdminOrdersViewModel

- Опис: оброблює інформацію та відправляє до AdminOrdersView.
- Атрибути: currentOrder, orders.
- Методи:

  getOrders() – оброблює замовлення з бази

Структура Order

- Опис: задає необхідні типи даних, які зберігає у массиві.
- Атрибут: id, userID, positions, date, status, cost, representation.
- Методи: відсутні.

Перелік OrderStatus

- Опис: перелік різних статусів замовлення
- Атрибути: відсутні.
- Методи: відсутні.

Клас AddProductView.

- Опис: відповідає за відображення екрану додавання продукції до бази.
- Атрибути: showImagePicker, image, title, price, description, dismiss.
- Методи: функція додавання відбувається в тілі кнопки зберігання.

Система User

Структура MainTabBar.

- Опис: відповідає за панель доступу до корзини, профілю та каталогу .
- Атрибути: user, body(UI).
- Методи: відсутні.

Клас CatalogView

- Опис: відображає колекцію продукції.
- Атрибути: layoutForPopular, layoutForPizza, viewModel, body.
- Методи: відсутні.

Клас CatalogViewModel

- Опис: отримує інформацію для усіх клітин колекції, .
- Атрибути: popularProducts, pizzas.
- Метод: getProducts() – отримує продукцію

Структура Cell

- Опис: осередок коллекції.
- Атрибут: product, body(UI)
- Методи: відсутні.

Клас ProductDetailView

- Опис: відображення: ціни, опису.
- Атрибути: viewModel, size, count, body.
- Методи: відсутні.

Клас ProductDetailViewModel

- Опис: отримує дані для Продукції.
- Атрибути: product, sizes, count, image.
- Методи:

  getPrice() – рахує ціну за розмір товару,

  getImage() – завантажує картинку товару.

Структура Product

- Опис: модель продукції.
- Атрибути: id, title, imageUrl, price, descript, representation.
- Методи: відсутні

Структура Position

- Опис: модель позицій.
- Атрибути: id, product, count, cost, representation.
- Методи: відсутні.

Структура PositionCell

- Опис: осередок колекції позицій.

- Атрибути: position, body.
- Методи: відсутні.

Клас CartView

- Опис: відображення екрану корзини.
- Атрибути: viewModel, body.
- Методи: відсутні.

Клас CartViewModel

- Опис: отримує дані з попереднього екрану Каталог та відправляє до CartView.
- Атрибути: positions, cost.
- Методи: addPosition().

Клас ProfileView

- Опис: Відображає дані про користувача, його поточні замовлення.
- Атрибути: viewModel, isAvaAlertPresented, isAuthViewPresented, isQuitAlertPresented, body.
- Методи: відсутні.

Клас ProfileViewModel

- Опис: отримує дані та відправляє до View.
- Атрибути: profile, orders.
- Методи:

  getOrders(). – отримує замолення

  setProfile(). – додає нові дані у базу

  getProfile(). – отримує дані з бази.

Структура UserInfo

- Опис: модель даних для Profile.
- Атрибути: id, name, phone, address, representation.
- Методи: відсутні.

2.2.3 Динамічна модель програми

Динамічна модель системи – сукупність співвідношень, що визначають вихід системи в залежності від входу та стану системи. Динамічна модель відтворює зміни об'єкта, що відбуваються з часом або особливості функціонування об'єкта. Динамічні моделі називають також функціональними [9].

Для представлення даної моделі будуть використані діаграми послідовностей.

На рисунках 2.19 – 2.22 представлено діаграми послідовності основних механік.

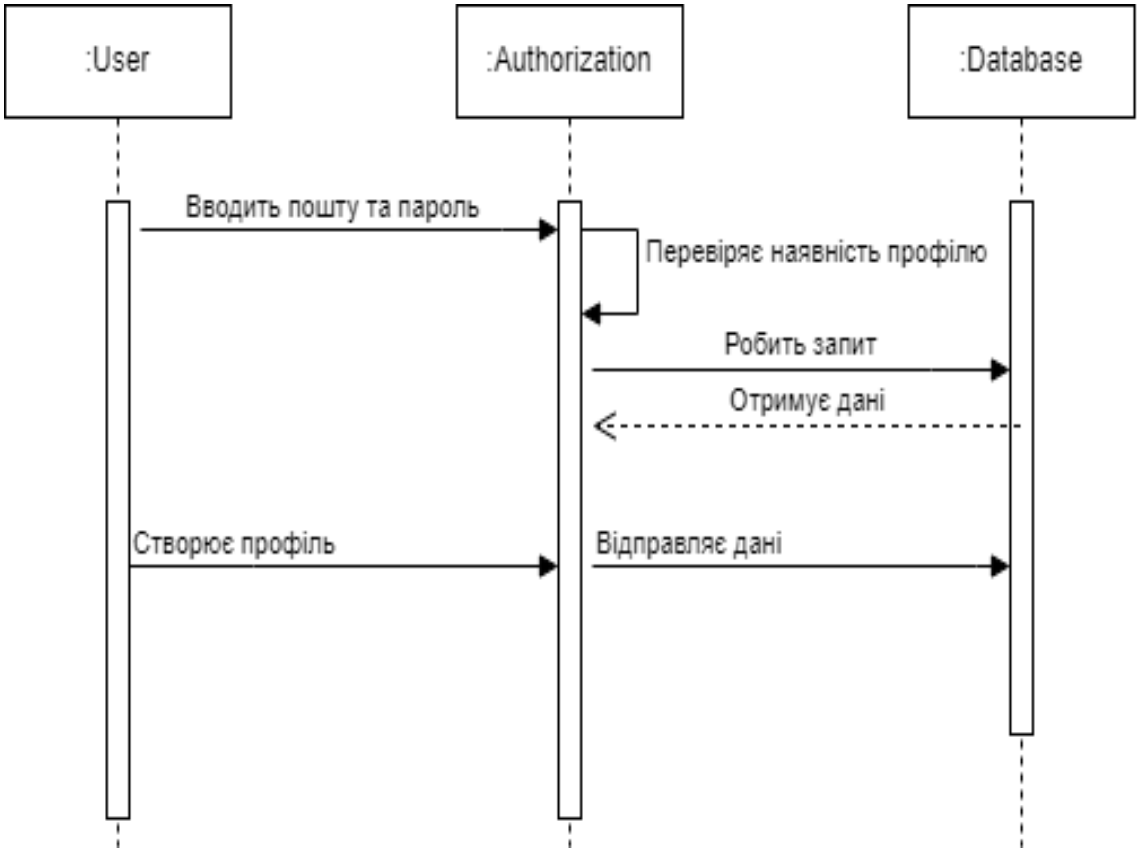

Рисунок 2.19 – Діаграма послідовності ігрової логіки в цілому

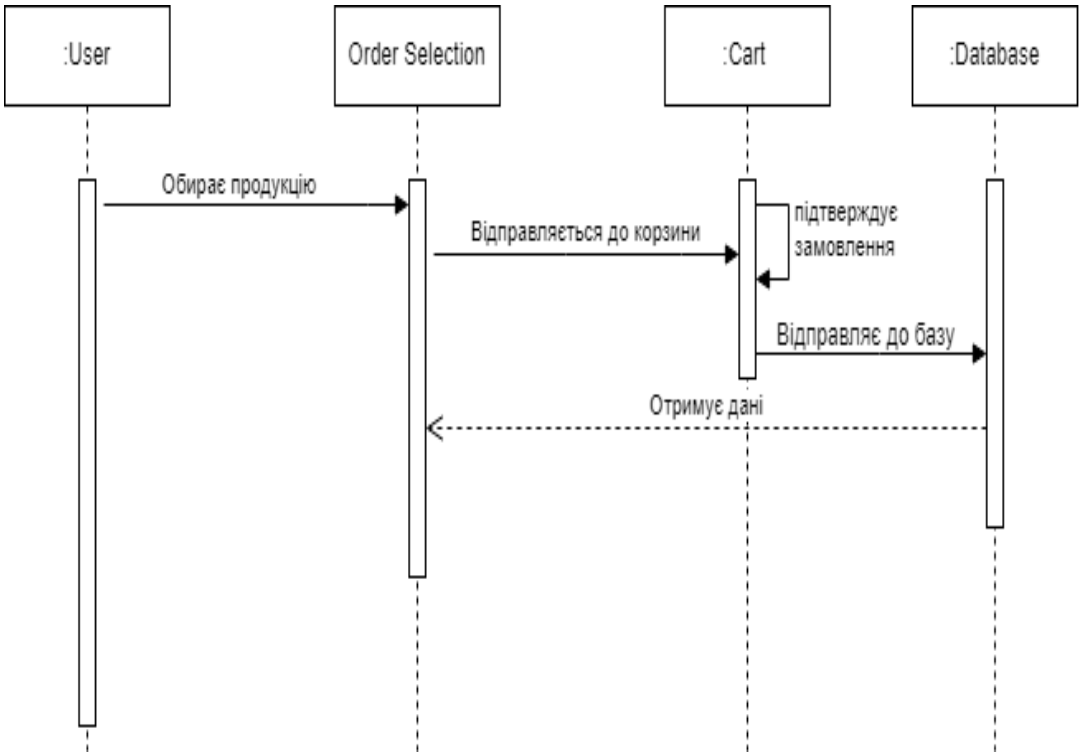

Рисунок 2.20 – Діаграма послідовності процесу створення серверу

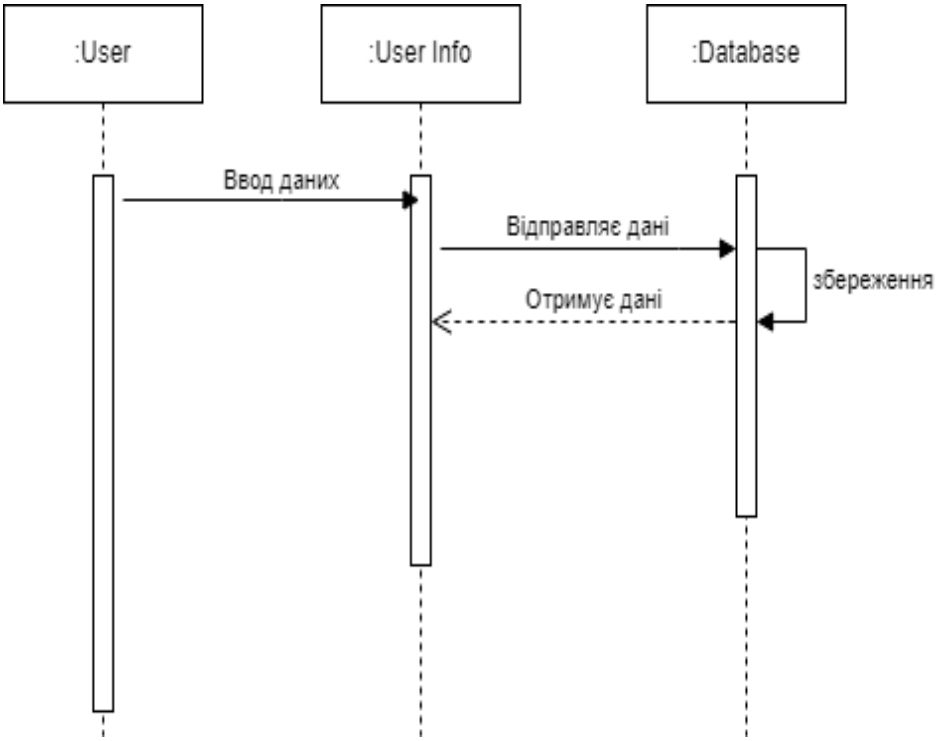

Рисунок 2.21 – Діаграма послідовності процесу підключення

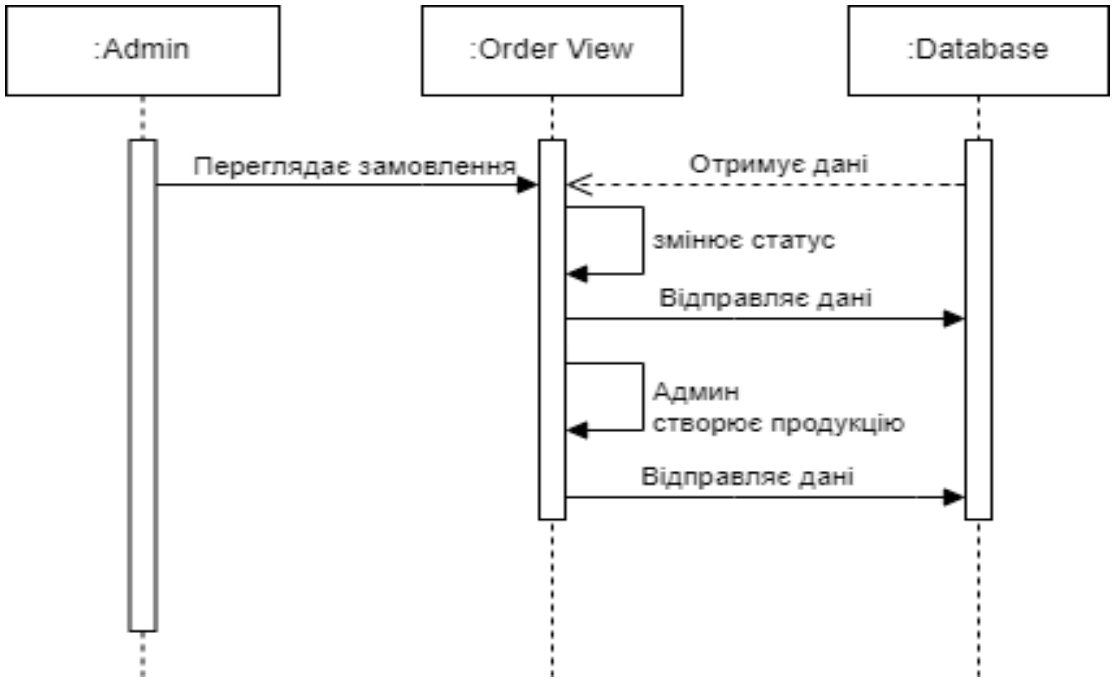

Рисунок 2.22 – Діаграма послідовності процесу обміну даними с гравця з сервером

2.2.4 Логічна модель даних

Логічна модель даних – це вдосконалена версія концептуальної моделі даних. Логічна модель даних конкретніша і враховує деталі реалізації. Вона включає всі сутності, атрибути, ключі і взаємозв'язки, які представляють бізнес-інформацію і визначають бізнес-правила.

На рисунках 2.23 та 2.24 представлені логічні моделі систем Admin Sestem та User System.

У таблицях 2.1 – 2.6 та 2.7 – 2.22. представлені логічні моделі цих систем у табличному форматі.

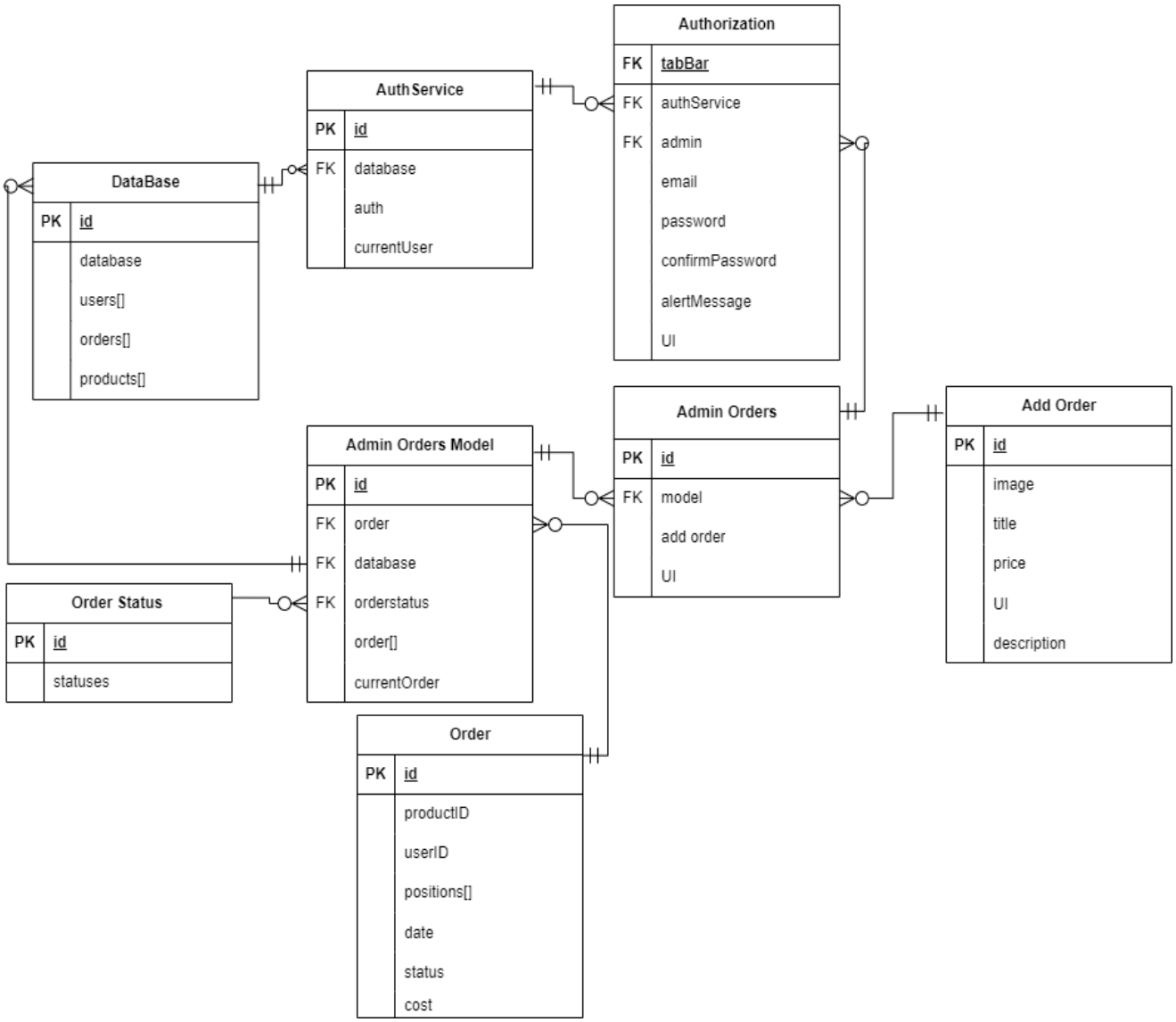

Рисунок 2.23 – Логічна модель даних системи Admin

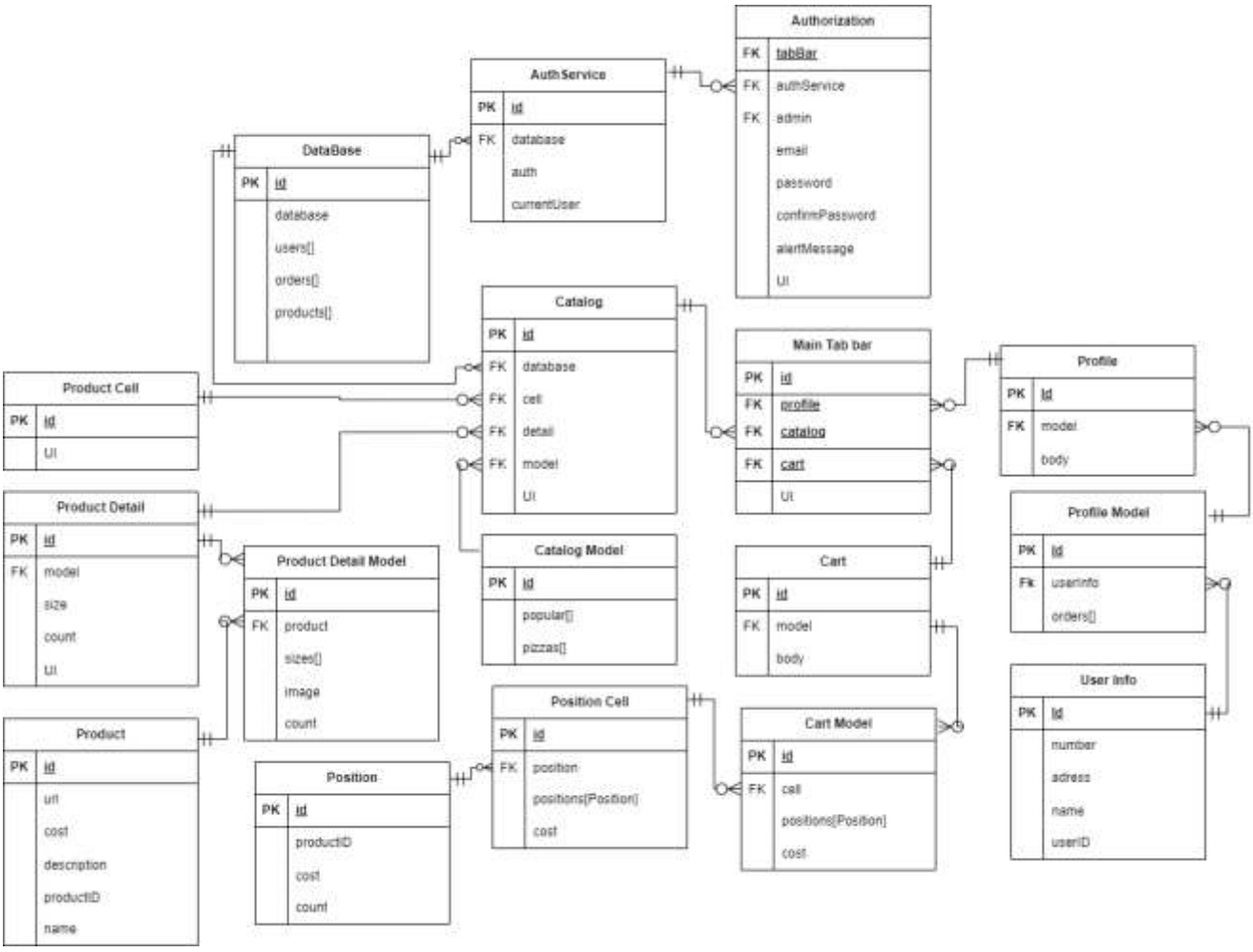

Рисунок 2.24 – Логічна модель даних системи User

Таблиця 2.1 – Таблиця Authorization систем Admin та User System

Назва поля	Тип даних	Ключ
authService	int	FK
admin	int	FK
email	string	
password	string	
confirmPassword	string	
alertMessage	string	
body(UI)	some view	

Таблиця 2.2 – Таблиця Auth Service систем Admin System та User System

Назва поля	Тип даних	Ключ
1	2	3
id	int	PK
database	int	FK
auth	Auth	
currentUser	User?	

Таблиця 2.3 – Таблиця DataBase систем Admin System та User System

Назва поля	Тип даних	Ключ
id	int	PK
database	StorageService	
users	CollectionReference[string]	
orders	CollectionReference[string]	
products	CollectionReference[string]	

Таблиця 2.4 – Таблиця AdminOrders системи Admin System

Назва поля	Тип даних	Ключ
id	int	PK
model	AdminOrdersViewModel[int]	FK
addOrder	int	FK
Body(UI)	some View	

Таблиця 2.5 – Таблиця Add Order системи Admin System

Назва поля	Тип даних	Ключ
id	int	PK
image	UIImage	
title	String	
price	Int?	
description	string	

Таблиця 2.6 – Таблиця Admin Orders Model системи Admin System

Назва поля	Тип даних	Ключ
id	int	PK
order	int	FK
database	int	FK
order status	int	FK
order	[Order]	
current order	order	

Таблиця 2.7 – Таблиця Order системи Admin System

Назва поля	Тип даних	Ключ
id	int	PK
productID	string	
userID	string	
positions	[Position]	
date	Date	
status	string	
cost	int	

Таблиця 2.8 – Таблиця Order Status(enum) системи Admin System

Назва поля	Тип даних	Ключ
id	int	PK
case: new	string	
case: cooking	string	
case: delivery	string	
case: completed	string	
case: cancel	string	

Таблиця 2.9 – Таблиця Main Tab Bar системи User System

Назва поля	Тип даних	Ключ
id	int	PK
profile	int	FK
catalog	int	FK
cart	int	FK
Body(UI)	some View	

Таблиця 2.10 – Таблиця Catalog системи User System

Назва поля	Тип даних	Ключ
id	int	PK
database	int	FK
cell	int	FK
detail	int	FK
model	int	FK
Body(UI)	some View	

Таблиця 2.11 – Таблиця Catalog Model системи User System

Назва поля	Тип даних	Ключ
id	int	PK
popular	[Product]	
pizzas	[Product]	

Таблиця 2.12 – Таблиця Product Cell системи User System

Назва поля	Тип даних	Ключ
id	int	PK
Body(UI)	some View	

Таблиця 2.13 – Таблиця Product Detail системи User System

Назва поля	Тип даних	Ключ
id	int	PK
model	int	FK
size	string	
count	int	
body(UI)	some View	

Таблиця 2.14 – Таблиця Product Detail Model системи User System

Назва поля	Тип даних	Ключ
id	int	PK
product	int	FK
sizes	[string]	
image	UIImage	
count	int	

Таблиця 2.15 – Таблиця Product системи User System

Назва поля1	Тип даних2	Ключ3
id	int	PK
url	string	
cost	int	
description	string	
productID	string	
name	string	

Таблиця 2.16 – Таблиця Position системи User System

Назва поля	Тип даних	Ключ
1	2	3
id	int	PK
product	int	FK
productID	string	
cost	int	
count	int	

Таблиця 2.17 – Таблиця Position Cell системи User System

Назва поля	Тип даних	Ключ
id	int	PK
position	int	FK
position	[Position]	
cost	int	

Таблиця 2.18 – Таблиця Cart системи User System

Назва поля	Тип даних	Ключ
id	int	PK
model	int	FK
Body(UI)	some View	

Таблиця 2.19 – Таблиця Cart Model системи User System

Назва поля	Тип даних	Ключ
id	int	PK
cell	int	FK
positions	[Position]	
cost	int	

Таблиця 2.20 – Таблиця Profile системи User System

Назва поля	Тип даних	Ключ
id	int	PK
model	int	FK
Body(UI)	some View	

Таблиця 2.21 – Таблиця Profile Model системи User System

Назва поля	Тип даних	Ключ
id	int	PK
user Info	int	FK
orders	[Order]	

Таблиця 2.22 – Таблиця User Info системи User System

Назва поля	Тип даних	Ключ
id	int	PK
number	int	
address	string	
name	string	
userID	string	

2.3 Робочий проект

В ході виконання роботи було створено діаграми компонентів та розміщення, описано вибір середовища розробки, проведено тестування та випробування програмного забезпечення.

2.3.1 Обґрунтування вибору мови й системи програмування

Для написання додатку «Pizza Shop» було обрано наступні інструменти:

- мова програмування Swift,
- симулятор LLVM,
- система зборки Xcode,
- бібліотеки: SwiftUI, Firebase, Foundation.

Насамперед треба відзначити, що для реалізації мобільного додатку не використовувалися жодні програми або фреймворки, які дозволяють спростити процес розробки. Додаток розроблявся з нуля, тому програмний код є як логікою додатку, так і основою, яка виконує важливу роль.

Для реалізації мобільного додатку було вирішено обрати мову програмування Swift. Ця мова була обрана з трьох причин:

- По-перше, Swift є високопродуктивною мовою, що дозволяє використовувати ресурси пристрою максимально ефективно, що покращує швидкодію додатку та робить користувацький досвід комфортнішим.
- По-друге, Swift має обширну базу потужних бібліотек, що дозволяє зробити процес розробки швидшим і більш продуктивним.
- По-третє, Swift є офіційною мовою програмування для платформи iOS, що забезпечує оптимальну інтеграцію з екосистемою Apple та дозволяє створювати додатки з високим рівнем надійності та сумісності з різними пристроями Apple.

У якості симулятору було обрано LLVM. Вибір впав саме на цей симулятор за причини – він є безкоштовним, оптимізованим, підтримує версії екосистеми починаючи з iOS - 10.6, також є основним стимулятором для Swift.

Xcode – це ключовий інструмент для розробки мобільних додатків під iOS від Apple, який обрав завдяки його численним перевагам. Це офіційне інтегроване середовище розробки, що забезпечує повну сумісність з API і інструментами компанії. Xcode надає відмінну підтримку Swift, включаючи автодоповнення,

рефакторинг та інтеграцію з Swift Package Manager та Cocoa Pods , що значно спрощує процес розробки. Використання компілятора Clang(LLMV) забезпечує швидку компіляцію і високоякісні повідомлення про помилки, що допомагає швидко і ефективно виправляти проблеми.[4]

Вибір бібліотеки SwiftUI, Foundation, тому що це основні інструменти при розробці мобільних додатків під екосистему Apple. Для бази даних було підключено бібліотеку Firebase.

SwiftUI надає спосіб для створення інтерфейсу застосунку. Завдяки цій бібліотеці можливо робити сучасні та динамічні інтерфейси [4].

Foundation – це ключовий фреймворк, який входить до стандартної бібліотеки мови Swift і надає базові інструменти для розробки додатків під платформи iOS, macOS, watchOS і tvOS. Цей фреймворк включає різноманітні класи, структури та функції, які забезпечують роботу з даними, файлами, мережами, рядками, колекціями та іншими ключовими аспектами розробки [4].

Firebase – це платформа для розробки мобільних і веб-додатків, яка належить компанії Google. Вона включає в себе різноманітні інструменти і сервіси для зберігання даних в реальному часі, аутентифікації користувачів, облачних зберігань, аналітики, тестування, розсилки повідомлень.

2.3.2 Розробка діаграми компонентів

Діаграма компонентів – статична структурна діаграма, показує розбиття програмної системи на структурні компоненти та зв`язки (залежності) між компонентами. У фізичних компонентів можуть виступати файли, бібліотеки, модулі, виконувані файли, пакети [10].

На рисунку 2.25 наведена діаграма компонентів, котра відображає основну суть системи та дозволяє зрозуміти зв’язки між головними елементами.

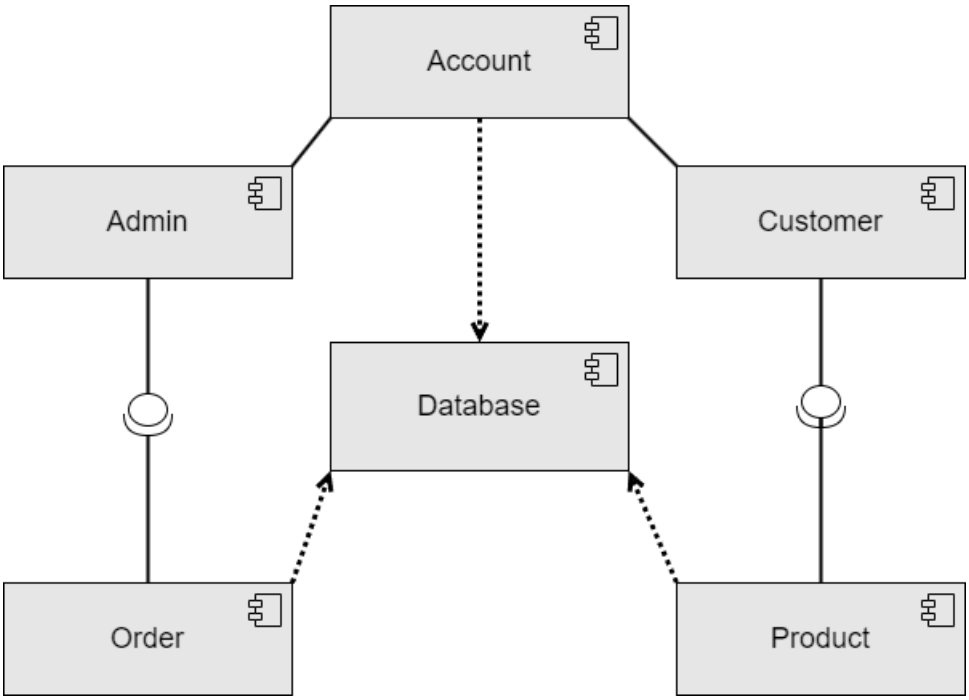

Рисунок 2.25 – Діаграма компонентів

2.3.3 Реалізація та тестування

У ході виконання кваліфікаційного проекту було проведено тестування методом «функціонального тестування».

Функціональне тестування – це тип тестування програмного забезпечення, який перевіряє функціональність програмної системи або програми. Він зосереджений на забезпеченні того, щоб система працювала відповідно до визначених функціональних вимог і відповідала передбачуваним потребам.

Метою функціонального тестування є перевірка функцій, можливостей і взаємодії системи з різними компонентами. Це включає тестування вхідних і вихідних даних програмного забезпечення, маніпулювання даними, взаємодію з користувачем. Функціональне тестування стосується лише перевірки того, чи система працює належним чином.

Під час тестування додатку, перевірялися основні вимоги до проекту, котрі описані у пункті 1.8, а також деякі вимоги, котрі описані у пункті 1.7. У даному розділі наведено лише ті протестовані вимоги, котрі можна підтвердити графічно.

Тест 1. Запуск екрану «Авторизації».

Вхідні дані: запуск застосунку.

Вихідні дані: відображення екрану «Авторизації».

Результат тестування: додаток запускається; користувацький UI відображається.

Рисунок з результатом тестування: рисунок 2.26.

Примітка: якщо користувача немає в базі, він отримує помилку на рисунку 2.27.

Рисунок 2.26 – екран Авторизації

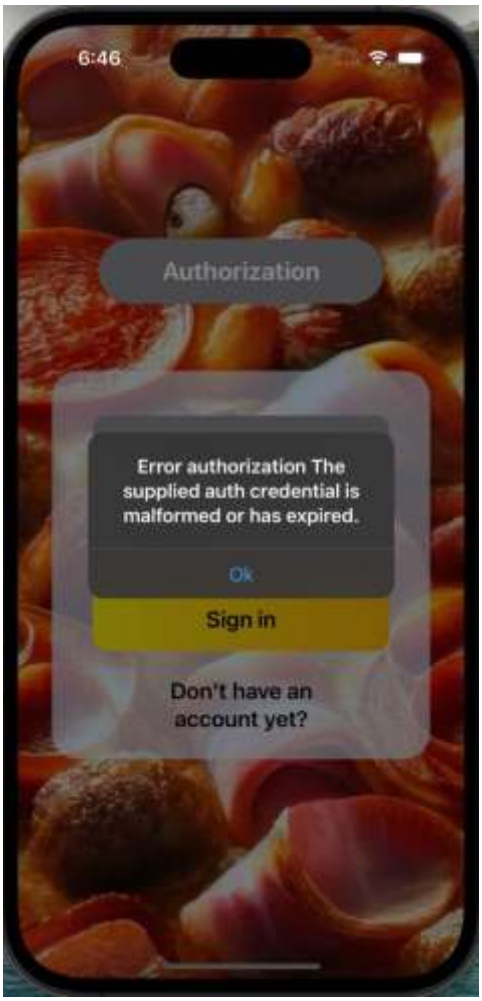

Рисунок 2.27 – Помилка входу

Тест 2. Створення облікового запису.

Вхідні дані: екран «Авторизації» відображає додаткове поле повторного вводу паролю та вже готовий до нових даних.

Вихідні дані: додаткові функції відображуються; користувач вводить нові дані.

Результат тестування: додаткове поле відображається; можливо ввести дані та пароль, перевіряється на збіг.

Рисунок з результатом тестування: 2.28.

Примітка: якщо пароль з другого поля не збігається, користувач отримує помилку як на рисунку 2.29.

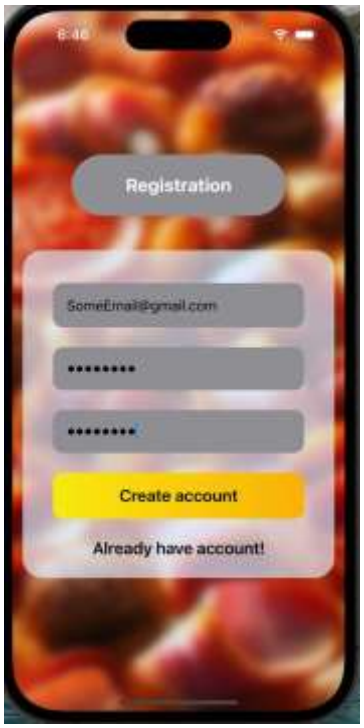

Рисунок 2.28 – Створення облікового запису

\

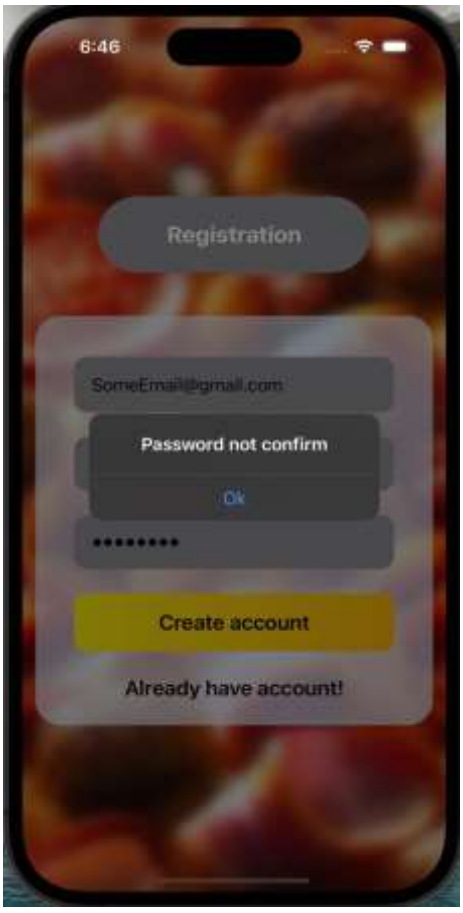

Рисунок 2.29 – Помилка підтвердження паролю

Тест 3. Відображення Каталогу.

Вхідні дані: вхід у обліковий запис.

Вихідні дані: відображення колекції «Каталогу».

Результат тестування: «Каталог» відображається, комірки присутні.

Рисунок з результатом тестування: 2.30.

Примітка: відсутня.

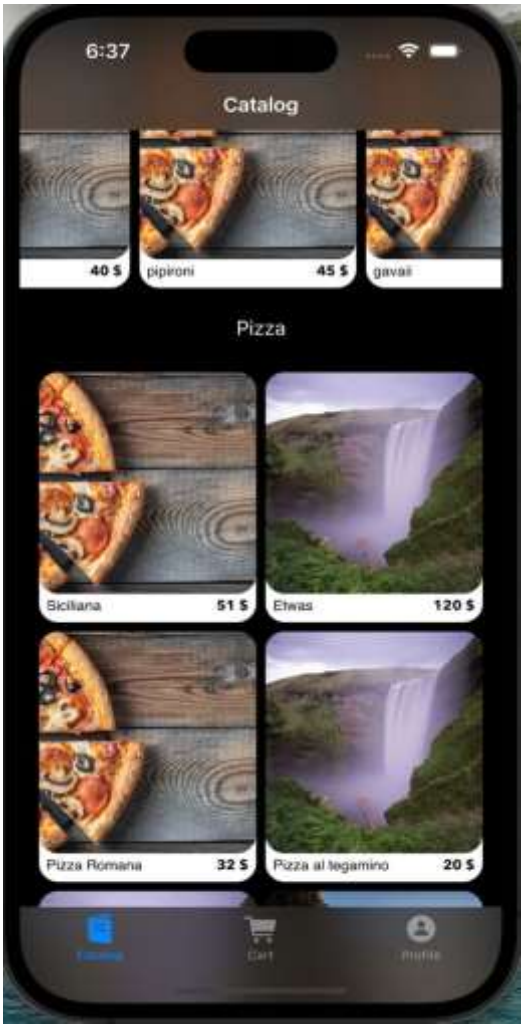

Рисунок 2.30 – Відображення Каталогу

Тест 4. Відображення опису продукту.

Вхідні дані: вибір продукції.

Вихідні дані: відображення вікна, де можна вибрати розмір замовлення та побачити опис.

Результат тестування: опис відкривається.

Рисунок з результатом тестування: 2.31.

Примітка: відсутня.

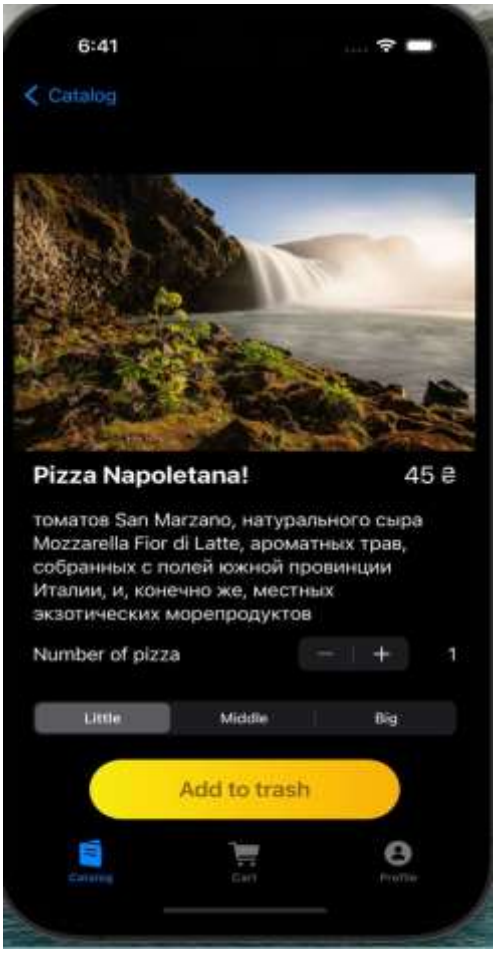

Рисунок 2.31 – Відображення опису Продукту

Тест 5. Підтвердження замовлення.

Вхідні дані: можливе замовлення відображається, очікується підтвердження.

Вихідні дані: замовлення відправляється до бази, відображається у Профілі та у Адміністратора.

Результат тестування: замовлення з’явилось у базі, відображається у профілі користувача та у Адміністратора.

Рисунок з результатом тестування: 2.32

Примітка: відсутня.

Рисунок 2.32 – відображення Корзини

Тест 6. Відображення Профілю.

Вхідні дані: перехід до екрану Профілю.

Вихідні дані: відображення Профілю та даних користувача.

Результат тестування: відображається Профіль та дані користувача.

Рисунок з результатом тестування: рисунок 2.33.

Примітка: якщо даних немає– рисунок 2.34.

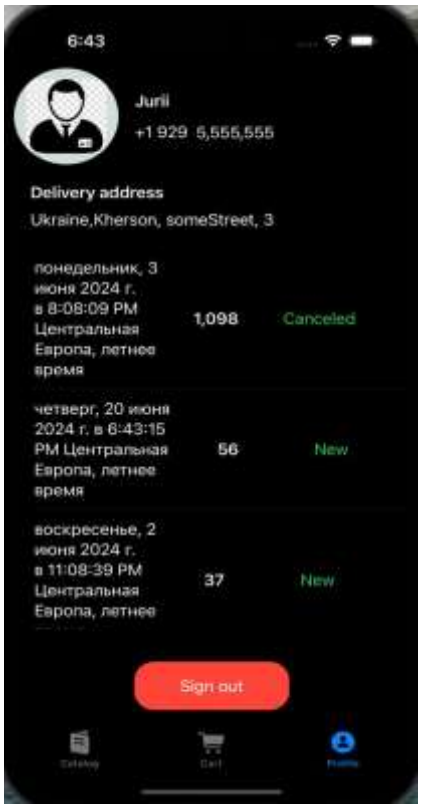

Рисунок 2.33 – відображення Профілю

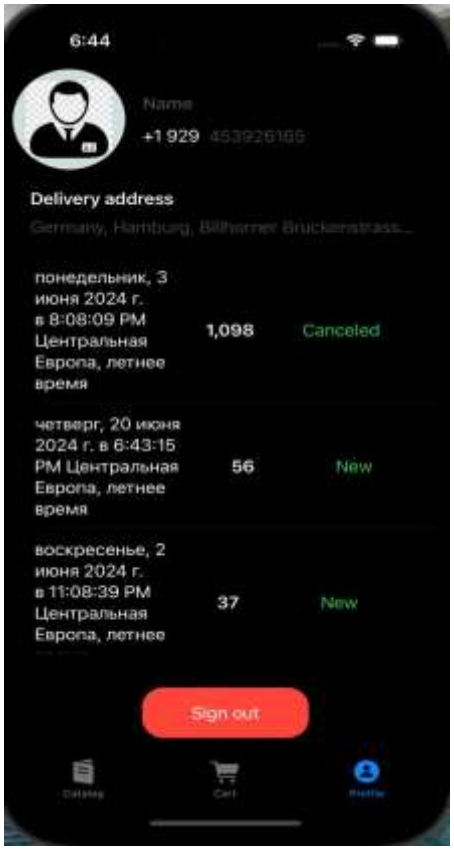

Рисунок 2.34 – відображення Профілю без наявності вхідних даних

Тест 7. Вихід з облікового запису.

Вхідні дані: натискається кнопка виходу, відкривається функція підтвердження дії.

Вихідні дані: здійснюється вихід.

Результат тестування: кнопка виходу натиснута, відкривається опція вибору.

Рисунки з результатами тестування: рисунок 2.35.

Примітка: відсутня.

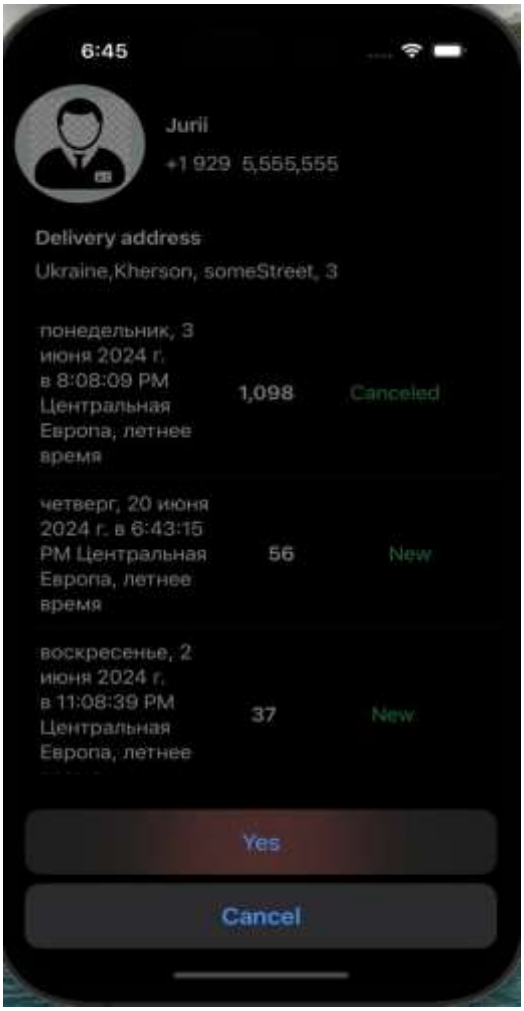

Рисунок 2.35 – Відображення опції підтвердження

Тест 8. Колекція замовлень Адміністратора.

Вхідні дані: отримання даних з бази даних.

Вихідні дані: відображення поточних замовлень.

Результат тестування: дані отримані, поточні замовлення відображаються.

Рисунок з результатом тестування: рисунок 2.36.

Примітка: відсутня.

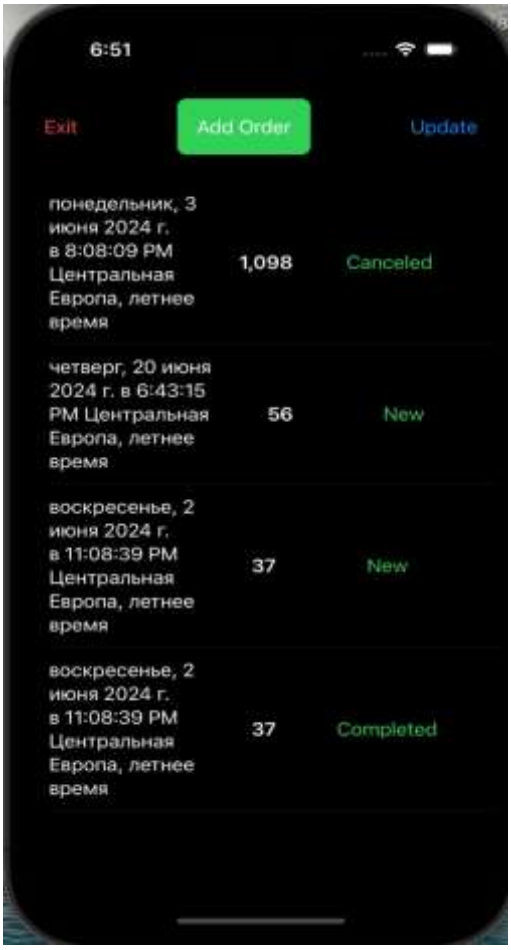

Рисунок 2.36 – Відображення Колекції

Тест 9. Зміна статусу Замовлення.

Вхідні дані: натиснутий перемикач, котрий відповідає за відкриття вікна.

Вихідні дані: відображення меню з певними опціями.

Результат тестування: вікно відкривається, зміни здійснюються.

Рисунок з результатом тестування: 2.37 – 2.38.

Примітка: відсутня.

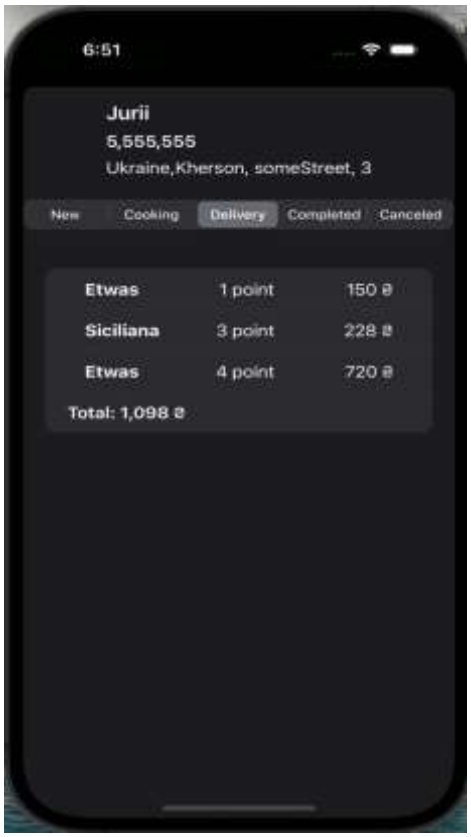

Рисунок 2.37 – Відображення Вікна зміни статусу

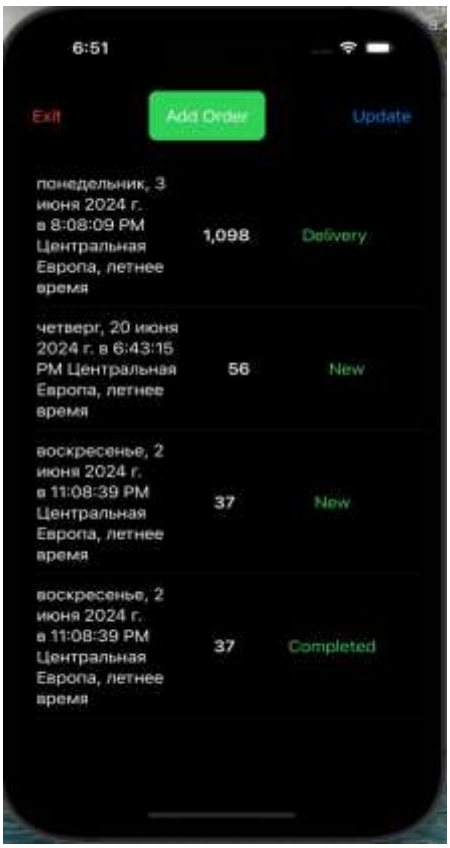

Рисунок 2.38 – Відображення змін

Тест10. Вікно додавання продукції.

Вхідні дані: натиснута кнопка «Add».

Вихідні дані: відображення полів для вводу даних.

Результат тестування: дані уведені, новий продукт відправляється до бази.

Рисунок з результатами тестування: 2.39

Примітка: відсутня.

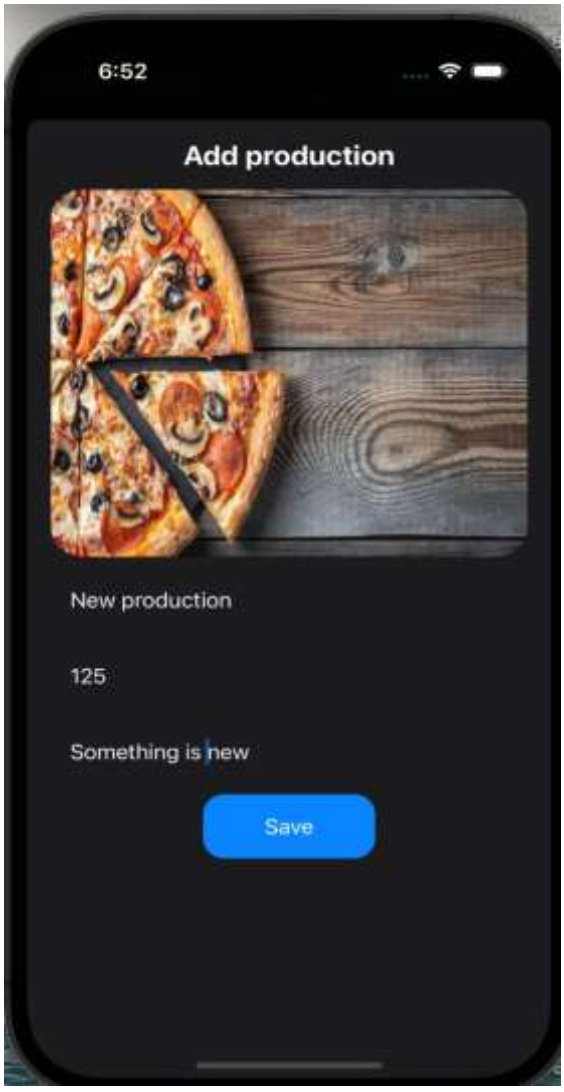

Рисунок 2.39 – Вікно додавання продукції

Для підсумування всіх тестів та наглядного відображення результатів тестування у таблиці 2.24 відображається Checklist.

Таблиця 2.24 – Checklist проведених тестів

№	Назва тесту	Результат	Наявність багу
1	Запуск екрану «Авторизації»	реалізовано	не виявлено
2	Створення облікового запису	реалізовано	не виявлено
3	Відображення Каталогу	реалізовано	виявлено
4	Відображення опису продукту	реалізовано	не виявлено
5	Підтвердження замовлення	реалізовано	не виявлено
6	Відображення Профілю	реалізовано	не виявлено
7	Вихід з облікового запису	реалізовано	не виявлено
8	Колекція замовлень Адміністратора	реалізовано	не виявлено
9	Зміна статусу замовлення	реалізовано	не виявлено
10	Вікно додавання продукції.	реалізовано	не виявлено

Результати проведення тестів

У ході тестування до було проведено десять тестів, які перевіряли роботу «Pizza Shop». Для підсумку, на рисунку 2.40 зображена діаграма проведених тестів.

Рисунок 2.40 – Діаграма проведених тестів

3 РЕЗУЛЬТАТИ РОЗРОБКИ ДОДАТКУ «Pizza Shop»

В результаті виконання кваліфікаційної роботи був розроблений додаток «Pizza Shop», який складається з двадцяти восьми файлів та приблизно з двух тисяч строк написаного виконуючого коду, що займає невелику кількість дискового простору, та надає такі функції:

- створення облікового запису;
- можливість продивитися колекцію товарів закладу;
- здійснити замовлення;
- ввести дані про користувача;
- дивитися за станами замовлення;
- створювати нові товари.

Проведено аналіз вже існуючих копій та клонів «Pizza Shop», обґрунтовано рішення розробки ще одного застосунку, застосувавши нову бібліотеку для створення додатку від Apple.

В процесі розробки було розроблено ескізний, технічний та робочий проекти, проведено налагодження та тестування, створений інтуїтивно зрозумілий інтерфейс, котрий робить додаток простим у використанні.

В рамках ескізного проекту було побудовано діаграму варіантів використання, описані специфікації варіантів використання, розроблені діаграми діяльності та концептуальна модель даних у вигляді діаграми сутність-зв’язок, представлено проект користувацького інтерфейсу.

В рамках технічного проекту програмного забезпечення було створено статичну модель програми (діаграма класів), описані специфікації класів, були побудовані діаграми послідовності до варіантів використання та була розроблена логічна модель даних.

В рамках робочого проекту програмного забезпечення було створено діаграму компонентів, описано вибір мови, проведено тестування та випробування додатку, яке підтвердило її працездатність.

Технічне завдання до «Pizza Shop» представлено у додатку А. Програма та методика випробувань «Pizza Shop» представлена у додатку Б. Розроблено інструкцію користувача, яка представлена у Додатку В. Опис програмного забезпечення «Pizza Shop» представлено у Додатку Г. А також текст програми наведений у Додатку Д.

Застосунок «Pizza Shop» був створений на мові програмування Swift з використанням Xcode для зборки додатку та LLMV для симуляції програмного коду у виконуючий файл.

Окрім, розроблених функцій, додаток має незавершені, котрі не увійшли до кінцевого результату. До цих функцій можна віднести спосіб оплати, вибір фото для Профілю з галереї, вікно з налаштуваннями. Також до готового додатку не було використано Grand Central Dispatch(GCD) для прискорення отримання даних з бази даних, це можливо побачити при загрузці колекції товарів користувача.

У майбутньому додаток можливо покращити та вдосконалити і в плані функцій, можливостей, і в візуальному плані. Наприклад можливо додати різні теми для застосунку, додати більше товарів та їх різноманітності.

4 ОХОРОНА ПРАЦІ

Охорона праці – це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних та лікувально-профілактичних заходів і засобів, спрямованих на збереження здоров'я та працездатності людини в процесі праці.

Умови праці – сукупність факторів виробничого середовища і трудового процесу, які впливають на здоров'я і працездатність людини в процесі її професійної діяльності.

Шкідливий виробничий фактор – виробничий фактор, вплив якого може призвести до погіршення стану здоров'я, зниження працездатності працівника. Охорона праці базується на законодавчих, директивних та нормативно-технічних документах. При управлінні охороною праці не повинні прийматись рішення та здійснювати заходи, що суперечать діючому законодавству, державним нормативним актам про охорону праці, стандартам безпеки праці, правилам та нормам охорони праці.

До основних функцій управління охороною праці належать:

- прогнозування і планування робіт, їх фінансування;
- організація та координація робіт;
- облік показників стану умов і безпеки праці;
- аналіз та оцінка стану умов і безпеки праці;
- контроль за функціонуванням системи управління охороною праці.
- Основні завдання управління охороною праці:
- навчання працівників безпечним методам праці та пропаганда питань охорони праці;
- забезпечення безпечності технологічних процесів, виробничого устаткування, будівель і споруд;
- нормалізація санітарно-гігієнічних умов праці;
- забезпечення працівників засобами індивідуального захисту;

- забезпечення оптимальних режимів праці та відпочинку;
- організація лікувально-профілактичного обслуговування;
- професійний добір працівників з окремих професій;
- удосконалення нормативної бази з питань охорони праці.

Охорона праці відіграє важливу роль, як суспільний чинник, оскільки, якими б вагомими не були трудові здобутки, вони не можуть компенсувати людині втраченого здоров'я, а тим більше життя – те і інше дається лише один раз. Тому цілком зрозуміло, що вивченню питань охорони праці приділяється серйозна увага. Представники багатьох наук вивчають проблеми створення безпечних та нешкідливих умов та засобів праці. Адже саме за таких обставин людина здатна працювати високопродуктивно, створювати необхідний матеріальний потенціал суспільства, добробут усіх громадян.

Основоположним законодавчим документом в галузі основи праці є Закон України «Про охорону праці» для якого поширюється на всі підприємства, установи і організації незалежно від форми власності та видів діяльності, на усіх громадян які працюють, а також залучені до праці на цих підприємствах. Все це затверджено у Законі України про «Охорону праці», який був прийнятий Верховною Радою України 14 жовтня 1992 року. Цей Закон визначає основні положення щодо реалізації конституційного права працівників на охорону їх життя і здоров'я у процесі трудової діяльності, на належні, безпечні і здорові умови праці, регулює за участю відповідних органів державної влади відносини між роботодавцем і працівником з питань безпеки, гігієни праці та виробничого середовища і встановлює єдиний порядок організації охорони праці в Україні.

4.1 Аналіз потенційно небезпечних і шкідливих факторів у приміщенні

Класифікація шкідливих та небезпечних виробничих факторів наведена у таблиці 4.1.

Таблиця 4.1 Класифікація шкідливих та небезпечних виробничих факторів

Види потенційно можливих небезпечних і шкідливих виробничих факторів	Наявність факторів
1	2
1. Фізичні небезпечні і шкідливі виробничі фактори:	
– шум,	+
– вібрація,	-
– підвищене виділення тепла,	-
– підвищена вологість,	-
– наявність електромагнітних полів,	+
– наявність електростатичних полів,	+
– небезпека вибуху,	-
– небезпека виникнення пожежі,	+
– небезпека ураження електрострумом,	+
– підвищена загазованість,	-
– відкрито рухаються й обертаються частини машин.	-
2. Хімічні небезпечні і шкідливі виробничі фактори:	
– токсичні речовини: пари ртуті, пари свинцю і його сполук та ін.,	-
– подразнюючі речовини: пральні засоби, кислоти, луги, роз-чинники, кислі гази,	-
– канцерогенні речовини: сполуки алюмінію, бензин, гас, 3,4-бензапирен (міститься у вихлопних газах автомобілів).	-
3. Біологічні небезпечні і шкідливі виробничі фактори:	
– мікроби, віруси, бактерії,	+
– комахи,	+
– гриби, найпростіші,	-
– інші мікроорганізми.	-
4. Психофізіологічні небезпечні і шкідливі виробничі фактори:	+
– малорухомість роботи,	+
– фізична утома,	+
– нервова напруга,	+
– напруга аналізаторів (зорового-слухового-тактильного)	+
– напруга рук, ніг, спини	

Розглянемо основні вимоги до приміщень, де встановлені комп'ютери. Залежно від орієнтації вікон рекомендується наступне забарвлення стін і підлоги приміщення:

- вікна орієнтовані на південь - стіни зеленувато-блакитного або світло блакитного кольору; підлога – зелений;

- вікна орієнтовані на північ - стіни світло-оранжевого або оранжево-жовтого кольору; підлога – червонувато-оранжевий;

- вікна орієнтовані на схід – стіни жовто-зеленого кольору; підлогу зелене або червонувато-оранжевий;

- вікна орієнтовані на захід - стіни жовто-зеленого або голубувато-зеленого кольору; підлога – зелена або червонувато-оранжевий.

Освітлення приміщень обчислювальних центрів повинно бути змішаним.

Вимоги до освітленості в приміщеннях, де встановлені комп'ютери, наступні: при виконанні зорових робіт високої точності загальна освітленість повинна складати 300 лк, а комбінована - 750 лк; аналогічні вимоги при виконанні робіт середньої точності – 200 і 300 лк відповідно.

4.1.1 Електробезпека

У приміщенні по розрахунках з побутовими і промисловими споживачами в обчислювальному центрі виробничого управління теплової енергії на кожному робочому місці використовується персональний комп’ютер та інші електричні прилади, які повинні бути встановлені дотримуючись вимог Правил влаштування електроустановок (ПВЕ - 86). Вимоги електробезпеки встановлені електронно-обчислюванні машини і персональні комп'ютери (ПК) відображені у ДНАОП 0.00-1.31-99. Відповідно до цього нормативного документу під час проектування систем електропостачання, монтажу основного електрообладнання та електричного освітлення будівель та приміщень для ПК необхідно дотримуватись вимог Правил влаштування електроустановок (ПВЕ), стандартів використання електрообладнання, Правил пожежної безпеки в Україні та інших нормативних документів, що стосуються штучного освітлення і електротехнічних пристроїв, а також вимог нормативно-технічної експлуатаційної документації заводу

виробника. ПК, периферійні пристрої ПК та устаткування для обслуговування, ремонту та налагодження ПК, інше устаткування, електропроводи та кабелі за виконанням та ступенем захисту мають відповідати класу зони за ПВЕ, мати апаратуру захисту від струму короткого замикання та інших аварійних режимів.

Під час монтажу та експлуатації ліній електромережі необхідно повністю унеможливити виникнення електричного джерела згорання внаслідок короткого замикання та перевантаження проводів, обмежувати застосування проводів з легкозаймистою ізоляцією і, за можливістю, перейти на негорючу ізоляцію.

Лінія електромережі для живлення ПК, периферійних пристроїв ПК та устаткування для обслуговування, ремонту та налагодження ПК виконується як окрема групова три провідна мережа, шляхом прокладання фазового, нульового робочого та нульового захисного провідників. Нульовий захисний провідник використовується для заземлення (занулення) електроприймачів і прокладається від стійки групового розподільчого щита, розподільчого пункту до розеток живлення.

Усі провідники повинні відповідати номінальним параметрам мережі та навантаження, умовам навколишнього середовища, умовам розподілу провідників, температурному режиму та типам апаратури захисту, вимогам ПВЕ.

У приміщенні, де одночасно експлуатується або обслуговується більше п'яти персональних ПК, на помітному та доступному місці встановлюється аварійний резервний вимикач, який може повністю вимкнути електричне живлення приміщення, крім освітлення.

ПК, периферійні пристрої ПК та устаткування для обслуговування, ремонту та налагодження ПК повинні підключатися до електромережі тільки з допомогою справних штепсельних з'єднань і електророзеток заводського виготовлення.

Штепсельні з'єднання та електророзетки крім контактів фазового та нульового робочого провідників повинні мати спеціальні контакти для підключення нульового захисного провідника. Необхідно унеможливити

з'єднання контактів фазових провідників з контактами нульового захисного провідника.

Неприпустимим є підключення ПК, периферійних пристроїв ПК та устаткування для обслуговування, ремонту та налагодження ПК до звичайної двопровідної електромережі, в тому числі – з використанням перехідних пристроїв.

Штепсельні з'єднання та електророзетки для напруги 12 В та 36 В за своєю конструкцією повинні відрізнятися від штепсельних з'єднань для напруги 127 В та 220 В. Окрім того вони мають бути пофарбовані в колір, який візуально значно відрізняється від кольору штепсельних з'єднань, розрахованих на напругу 127 В та 220 В.

Індивідуальні та групові штепсельні з'єднання та електророзетки необхідно монтувати на негорючих або важкогорючих пластинах з урахуванням вимог ПВЕ та Правил пожежної безпеки в Україні.

Електромережу штепсельних розеток для живлення персональних ПК, периферійних пристроїв ПК та устаткування для обслуговування, ремонту та налагодження ПК при розташуванні їх уздовж стін приміщення прокладають по підлозі поряд зі стінами приміщення, як правило, в металевих трубах і гнучких металевих рукавах з відводами відповідно до затвердженого плану розміщення обладнання та технічних характеристик обладнання.

При розташуванні в приміщенні за його периметром до 5 персональних ПК, використанні трипровідникового захищеного проводу або кабелю в оболонці з негорючого або важкогорючого матеріалу дозволяється прокладання їх без металевих труб та гнучких металевих рукавів.

Електромережу штепсельних розеток для живлення персональних ПК, периферійних пристроїв ПК та устаткування для обслуговування, ремонту та налагодження ПК при розташуванні їх у центрі приміщення, прокладають у каналах або під знімною підлогою в металевих трубах або гнучких металевих рукавах. При цьому не дозволяється застосовувати провід і кабель в ізоляції з

вулканізованої гуми та інші матеріали, що містять сірку. Відкрита прокладка кабелів під підлогою забороняється.

Металеві труби та гнучкі металеві рукави повинні бути заземлені. Заземлення повинно відповідати вимогам ДНАОП 0.00-1.21-98 «Правила безпечної експлуатації електроустановок споживачів».

Отвори в плитах для прокладання кабелів електроживлення виконуються безпосередньо в місцях встановлення устаткування відповідно до затвердженого технологічного плану розміщення устаткування та його технічних характеристик.

Для підключення переносної електроапаратури застосовують гнучкі проводи в надійній ізоляції. Тимчасова електропроводка від переносних приладів до джерел живлення виконується найкоротшим шляхом без заплутування проводів у конструкціях машин, приладів та меблях. Доточувати проводи можна тільки шляхом паяння з наступним старанним ізолюванням місць з'єднання.

Є неприпустимими:

– експлуатація кабелів та проводів з пошкодженою або такою, що втратила

– захисні властивості за час експлуатації, ізоляцією;

– залишення під напругою кабелів та проводів з неізольованими провідниками;

– застосування саморобних подовжувачів, які не відповідають вимогам ПВЕ до переносних електропроводок;

– застосування для опалення приміщення нестандартного (саморобного) електронагрівального обладнання або ламп розжарювання;

– користування пошкодженими розетками, розгалужувальними та з'єднувальними коробками, вимикачами та іншими електроприладами, а також лампами, скло яких має сліди затемнення або випинання;

– підвішування світильників безпосередньо на струмопровідних проводах, обгортання електроламп і світильників папером, тканиною та іншими горючими матеріалами, експлуатація їх зі знятими ковпаками (розсіювачами);

– використання електроапаратури та приладів в умовах, що не відповідають вказівкам (рекомендаціям) підприємств-виготовлювачів.

4.1.2 Пожежна безпека

Залежно від особливостей виробничого процесу, крім загальних вимог пожежної безпеки, здійснюються спеціальні протипожежні заходи для окремих видів виробництв, технологічних процесів та промислових об’єктів. Для споруд та приміщень, в яких експлуатуються відеотермінали та ПК такі заходи визначені Правилами пожежної безпеки в Україні, ДНАОП 0.00-1.31.99 та іншими нормативними документами.

Для приміщень повинна бути визначена категорія з вибухопожежної і пожежної безпеки відповідно до стандартів визначення категорій приміщень і будинків за вибухопожежною та пожежною небезпеки, та клас зони згідно з ПВЕ. Відповідні позначення повинні бути нанесені на вхідні двері приміщення.

Будівлі і ті їх частини, в яких розташовуються ПК, повинні мати не нижче II ступеня вогнестійкості. Якщо відповідно до СНиП 2.09.02-85 ці приміщення повинні бути відокремленими від приміщень іншого призначення протипожежними стінами, то межа їх вогнестійкості визначається відповідно до стандартів вогнегасників.

Неприпустимим є розташування приміщень категорій А і Б, а також виробництв з мокрими технологічними процесами поряд з приміщенням, де розташовуються ПК.

Приміщення для розрахунків зі споживачами за характеристикою речовин та матеріалів відноситься до категорії Д, тобто там знаходяться негорючі речовини та матеріали у холодному стані.

Сховища інформації слід розміщати у відокремлених приміщеннях, обладнаних негорючими стелажами і шафами. Фальшпідлога повинна бути виготовлена з негорючих матеріалів. Міжпідлоговий простір під знімною

підлогою має бути оснащений системою автоматичної пожежної сигналізації та засобами пожежогасіння відповідно до вимог.

Звукопоглинальне облицювання стін та стель слід виготовляти з негорючих або важкогорючих матеріалів.

Приміщення повинно бути оснащене системою автоматичної пожежної сигналізації з димовими пожежними сповіщувачами та переносними вуглекислотними вогнегасниками з розрахунку 2 шт. на кожні 20 м2 площі приміщення.

Не рідше одного разу на квартал необхідно очищати від пилу агрегати та вузли, кабельні канали та простір між підлогами.

Для запобігання виникнення пожежі необхідно передбачити міри пожежної профілактики: дотримання протипожежних вимог при проектуванні й експлуатації систем вентиляції згідно стандартів; дотримання умов пожежної безпеки електроустановок згідно ПУЕ-84; наявність засобів оповіщення:

- пожежні повідомлювачі (ЛИПНУВ-1, ИП-105 2/1 і т.д.);
- установки пожежогасіння (АУП);
- інструкції з мір протипожежної безпеки, план евакуації людей і технічних засобів.

Для поліпшення умов пожежної безпеки в повинна бути встановлена підлога з непальних матеріалів, технологічно знімний; папір зберігається в металевій шафі; у наявності два вуглекислотних вогнегасники типу ОУ-5, а також два димових датчики; у машинному залі систематично проводиться вологе збирання і вентилювання приміщення.

4.1.3 Освітлення

Освітлення у приміщенні, де використовується персональний комп’ютер, може здійснюватися природним і штучним освітленням. Приміщення з ПК повинні мати природне і штучне освітлення відповідно до стандартів природного

та штучного освітлення. Освітлення може здійснюватися природним і штучним світлом. При недостатності природного освітлення використовується змішане освітлення. Останнє є освітленням, при якому в світлий час доби використовується одночасно природне і штучне світло.

Штучне освітлення даного приміщення має бути обладнане системою загального рівномірного освітлення. У виробничих та адміністративно-громадських приміщеннях, де переважають роботи з документами, допускається вживати систему комбінованого освітлення (додатково до загального освітлення встановлюються світильники місцевого освітлення).

Загальне освітлення має бути виконане у вигляді суцільних або переривчатих ліній світильників, що розміщуються збоку від робочих місць (переважно зліва) паралельно лінії зору працівників. При розташуванні відеотерміналів ПК за периметром приміщення лінії світильників штучного освітлення повинні розміщуватися локально над робочими місцями. Для загального освітлення необхідно застосовувати світильники із розсіювачами та дзеркальними екранними сітками або віддзверкалювачами, укомплектовані високочастотними пускорегулювальними апаратами (ВЧ ПРА). Застосування світильників без розсіювачів та екранних сіток забороняється.

Як джерело світла при штучному освітленні повинні застосовуватися, як правило, люмінесцентні лампи типу ЛБ. Рівень освітленості на робочому столі в зоні розташування документів має бути в межах 300-500 лк. У разі неможливості забезпечити даний рівень освітленості системою загального освітлення допускається застосування світильників місцевого освітлення, але при цьому не повинно бути відблисків на поверхні екрану та збільшення освітленості екрану більше ніж 300 лк.

Для штучного освітлення приміщень варто використовувати люмінесцентні лампи, тому що в них висока світлова віддача (до 75 лм/Вт і

більш), тривалий термін служби (до 10000 годин), мала яскравість світної поверхні, близький до природного спектральний склад випромінюваного світла,

що забезпечує гарну передачу кольору. Найбільш прийнятними для дисплейних приміщень є люмінесцентні лампи ЛБ (білого світла) і ЛТБ (тепло-білого світла) потужністю 40, 80 Вт.

Для виключення засліплювання екранів дисплеїв прямими світловими потоками світильники загального освітлення розташовують збоку від робочого місця, паралельно лінії зору оператора і стіні з вікнами. Таке розміщення світильників дозволяє робити їхнє послідовне включення в залежності від величини природної освітленості і виключає роздратування очей смугами світла, що чергуються, і тіні, що виникають при поперечному розташуванні світильників.

4.1.4 Захист від шуму та вібрації

Широке промислове та побутове використання ПК актуалізувало питання охорони праці їхніх користувачів. Відомо, що шум несприятливо діє на слуховий аналізатор та інші органи та системи організму людини. Визначальне значення щодо такої дії має інтенсивність шуму, його частотний склад, тривалість щоденного впливу, індивідуальні особливості людини, а також специфіка виробничої діяльності. Шум, навіть і невеликий (50-60 дБА), створює значне напруження на нервову систему людини, надаючи негативну психологічну дію. Зі збільшенням рівня шуму до 70 дБА і вище, шум може спричиняти певний фізіологічний вплив на людину, приводячи до видимих змін в його організмі. Під дією шуму, який перевищує 85-90 дБА, в першу чергу знижується слухова чутливість на високих частотах. Рівень шуму в 120-130 дБА викликає больові відчуття. Звуки, що перевищують по своєму рівню цей поріг, можуть викликати не тільки больові відчуття, але й навіть пошкодження слухового апарату. При дії шуму більш, ніж 145 дБА можливий розрив барабанної перетинки.

Відповідно до стандартів при роботі з шумом рекомендуються такі діапазони звукового тиску всередині приміщень: для сну, відпочинку: 30-45 дБ – для розумової праці: 45-55 – для лабораторних досліджень, роботи з

персональним комп’ютером: 50-65 – для виробничих цехів, магазинів, гаражів : 56-70 дБ.

Ті види діяльності, у яких поєднується напружена розумова робота та інтенсивне використання комп’ютера характеризується відчутним впливом навіть незначних рівнів шуму. Цей вплив виражається у зниженні розумової працездатності, швидкій втомлюваності, послабленні уваги, появі головного болю та ін. На комп’ютеризованих робочих місцях основними джерелами шуму є вентилятори системного блоку, накопичувачі, принтери ударної дії. Для зниження рівнів шуму на робочих місцях рекомендується розмістити друкувальні пристрої ударної дії (струменеві, лазерні принтери тощо) в іншому приміщенні, або огородити їх звукоізолюючими екранами.

Для забезпечення нормованих рівнів шуму у виробничих приміщеннях та на робочих місцях застосовуються шумопоглинальні засоби, вибір яких обґрунтовується спеціальними інженерно-акустичними розрахунками.

Як засоби шумопоглинання повинні застосовуватися негорючі або важкогорючі спеціальні перфоровані плити, панелі, мінеральна вата з максимальним коефіцієнтом звукопоглинання в межах частот 31,5 і 8000 Гц, або інші матеріали аналогічного призначення, дозволені для оздоблення приміщень органами державного санітарно-епідеміологічного нагляду. Крім того, необхідно застосовувати підвісні стелі з аналогічними властивостями.

Основними заходами та засобами боротьби з шумом є: зниження рівнів шуму в джерелі його утворення (застосовується, як правило, в процесі проектування), використання звукопоглинаючих та звукоізолюючих засобів, раціональне планування виробничих приміщень та робочих місць.

Серед усіх видів механічного впливу для технічних об’єктів найбільш небезпечною є вібрація, яка викликає появу тріщин, руйнування, відмову роботи машин, пристроїв. Вібрація також шкідлива для людського організму, викликає порушення фізіологічного і функціонального стану людини, яке характеризується появою головної болі, онімінням пальців рук, болі у суглобах, появою судом,

безсоння, погіршенням зору, зміною реакції вестибулярного апарату, швидкою стомлюваністю.

Вплив вібрації на організм визначається наступними характеристиками: інтенсивністю, спектральним складом, тривалістю впливу, напрямком дії.

Основними заходами та засобами боротьби з вібрацією є: зниження вібрацій в джерелі появи шляхом зниження або усунення збуджуючих сил, використання індивідуальних засобів захисту, введення в коливальну систему додаткового пружного зв'язку, який запобіжить передачі енергії від коливального агрегату до основи або від коливальної основи до людини чи конструкції, які захищаються. Для зниження вібрації обладнання, пристрої необхідно встановлювати на спеціальні амортизуючи прокладки, передбачені нормативними документами.

Рівні вібрації під час виконання робіт з ПК у виробничих приміщеннях не повинні перевищувати допустимих значень, визначених в санітарних нормах вібрації робочих місць.

4.1.5 Виробничий мікроклімат

Приміщення повинно бути обладнане системами опалення, кондиціювання повітря або припливно-витяжною вентиляцією відповідно до стандартів вентиляції виробничих приміщень.

Параметри мікроклімату, іонного складу повітря, вміст шкідливих речовин на робочих місцях, оснащених ПК, повинні відповідати нормам мікроклімату виробничих приміщень, нормам робочої зони, нормам допустимих рівнів іонізації приміщень (таблиця 4.3).

Таблиця 4.2 – Нормовані параметри мікроклімату для приміщень з ВДТ та ППК

Пора року	Категорія робіт	Температура повітря, ˚С оптимальна	Відносна вологість повітря, % оптимальна	Швидкість руху повітря, м/с оптимальна
Холодна	Легка - 1а	22 –24	40 – 60	0,1

	Легка - 1б	21 – 23	40 – 60	0,1
Тепла	Легка - 1а	23 – 25	40 – 60	0,1
	Легка - 1б	22 – 24	40 – 60	0,2

Таблиця 4.3 – Рівні іонізації повітря приміщень при роботі на ВДТ та ППК

Рівні	Кількість іонів в 1 $см^3$ повітря	
	n^+	n^-
Мінімально необхідні	400	600
Оптимальні	1500 – 3000	3000 – 5000
Максимально допустимі	50000	50000

Для підтримки допустимих значень мікроклімату та концентрації позитивних та негативних іонів необхідно передбачати установки або прилади зволоження та/або штучної іонізації, кондиціювання повітря.

4.1.6 Організація робочого місця

Робоче місце – це місце постійного або тимчасового перебування працівника в процесі трудової діяльності. Організація

робочого місця користувача приміщення по розрахунках з побутовими і промисловими споживачами в обчислювальному центрі Виробничого управління теплової енергії повинна забезпечувати відповідність усіх елементів робочого місця та їх розташування ергономічним вимогам стандартів робочих місць при виконанні роботи сидячи.

Площа, виділена для одного робочого місця з відео терміналом або персональною ПК, повинна складати не менше 6 м , а обсяг – не менше 20 м .

Робочі місця з відео терміналами відносно світлових прорізів повинні розміщуватися так, щоб природне світло падало збоку, переважно зліва.

При розміщенні робочих місць необхідно дотримуватись таких вимог:

− робочі місця розміщуються на відстані не менше 1 м від стін зі світловими прорізами;

− відстань між бічними поверхнями відео терміналів має бути не меншою за 1,2 м;

− відстань між тильною поверхнею одного відео термінала та екраном іншого не повинна бути меншою 2,5 м;

− прохід між рядами робочих місць має бути не меншим 1 м.

Вимоги щодо відстані між бічними поверхнями відео терміналів та відстані між тильною поверхнею одного відео терміналу та екраном іншого враховуються також при розміщенні робочих місць з відео терміналами та персональними комп'ютерів в суміжних приміщеннях, з урахуванням конструктивних особливостей стін та перегородок.

Висота робочої поверхні столу для відео терміналу має бути в межах 680-800 мм, а ширина – забезпечувати можливість виконання операцій в зоні досяжності моторного поля.

Робочий стіл для відео терміналу, як правило, має бути обладнаним підставкою для ніг шириною не менше 300 мм та глибиною не менше 400 мм, з можливістю регулювання по висоті в межах 150 мм та кута нахилу опорної поверхні – в межах 20°. Підставка повинна мати рифлену поверхню та бортик на передньому краї заввишки 10 мм.

Робоче сидіння (стілець, крісло) користувача відео терміналу та персональної ПК повинно мати такі основні елементи: сидіння, спинку та стаціонарні або знімні підлокітники, також повинно бути підйомно-поворотним, таким, що регулюється за висотою, кутом нахилу сидіння та спинки, за відстанню спинки до переднього краю сидіння, висотою підлокітників. Поверхня сидіння має бути плоскою, передній край – заокругленим. Висота спинки сидіння має становити 300±20 мм, ширина – не менше 380 мм, радіус кривизни в горизонтальній площині – 400 мм. Кут нахилу спинки повинен регулюватися в межах 0-30° відносно вертикального положення. Відстань від спинки до переднього краю сидіння повинна регулюватись у межах 260-400 мм.

Екран відео термінала та клавіатура мають розташовуватися на оптимальній відстані від очей користувача, але не ближче 600 мм, з урахуванням розміру алфавітно-цифрових знаків та символів.

Клавіатуру слід розміщувати на поверхні столу або на спеціальній, регульованій за висотою, робочій поверхні окремо від столу на відстані 100-300 мм від краю, ближчого до працівника. Кут нахилу клавіатури має бути в межах 5-15°.

4.2 Розрахунок освітлення

Зробити розрахунок КПО попереднім методом для виробничої лабораторії з боковим одностороннім освітленням. Приміщення розташоване в 4 поясі світлового клімату. Довжина приміщення: 9м, глибина: 8м. Розрахункова точка А знаходиться від зовнішньої стіни на відстані: 1 м. Відстань від робочої поверхні до верха вікон: 4м.

Нормоване значення коефіцієнту бокового природного освітлення для III розряду зорової роботи III світлового поясу $e_н^{III}$ – 1.5%. У зв'язку з тим, що приміщення знаходиться в IV світловому поясі, КПО буде визначено за формулою:

$$e_н^{IV} = e_н^{III} mc \qquad (4.1)$$

де m = 0,9 – коефіцієнт світлового клімату;

C = 0,7 – коефіцієнт сонячності клімату.

$$e_н^{IV} = 1.5x0.9x0.7 = 0.945\%$$

Загальний коефіцієнт світло пропускання визначається за формулою:

$$\tau_0 = \tau_1 \tau_2 \tau_3 \tau_4 \tau_5 \qquad (4.2)$$

де τ_1 = 0,8 – подвійні склопакети;

τ_2 = 0,6 – Плетення подвійне дерев'яне;

τ_3 = 1 – Несущі конструкції відсутні;

$\tau_4 = 1$ – Сонце захисні конструкції відсутні

$\tau_5 = 1$ – Захисних сіток немає

τ_0=0.8x0.6x1x1x1=0,48

Площа світлових прорізів при боковому освітленні визначається за формулою:

$$S_б = \frac{S_п e_н K_з \eta_в K_б}{100 r_1 \tau_0} \quad (4.3)$$

де $S_п$ – площа підлоги приміщення,

$S_п = 9 \text{ x } 8 = 72 м^2$;

$K_З = 1,2$ – коефіцієнт запасу;

$\eta_в = 13$ – світлова характеристика вікон;

$к_б = 1$ – коефіцієнт, що враховує затінення вікон будинками, що стоять навпроти;

$r_1 = 1$- коефіцієнт, що враховує підвищення КПО при боковому освітленні завдяки світлу, відбитому від поверхонь приміщення і підстильного шару, що прилягає до будинку.

$$S_б = \frac{72 * 0,945 * 1,2 * 13 * 1}{100 * 1 * 0,48} = 22,113 м^2$$

Загальна ширина вікон знаходиться за допомогою наступної формули:

$$b_в = \frac{S_б}{h_в} \quad (4.4)$$

де $h_В = 3м$ – висота вікна

$$b_в = \frac{22,113}{3} = 7,371 м$$

Приймається 3 вікна, які мають ширину 2м.

ВИСНОВКИ

В результаті виконання кваліфікаційної роботи була досягнута поставлена мета та створений мобільний додаток з унікальним функціоналом, який має забезпечити зручність та ефективність користування.

Для досягнення мети кваліфікаційної роботи був проведений аналіз предметної області, виділені основні концептуальні класи. Був проведений аналіз вимог, побудована діаграма варіантів використання і визначена функціональність системи. Спроектовано інтуїтивно зрозумілий для користувача інтерфейс, який забезпечує зручне та приємне користування додатком. На основі аналізу предметної області та діаграми використання була спроектована структура класів системи.

В ході виконання проекту були покращені навички роботи з мовою програмування Swift, отримані навички роботи з графічним фреймворком SwiftUI та інтеграції з Firebase для зберігання даних у реальному часі. Також отримані навички в розробці мобільних додатків для платформи iOS.

У розробленому додатку виконана більша частина поставлених вимог. Додаток відповідає основним вимогам, що були визначені на початку роботи, та виконує всі основні функції, необхідні для забезпечення ефективного та приємного користування.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТИУРИ

1. Мобільний застосунок - url: https://uk.wikipedia.org/wiki/Мобільний_застосунок.
2. Типи мобільних додатків – url:https://training.qatestlab.com/blog/technical-articles/types-of-mobile-applications/.
3. Додатки доставки за рейтингом - url:https://stfalcon.com/uk/blog/post/on-demand-food-delivery-apps
4. Основи розробки на мові Swift. – Василій Усов видання №6,

Документація по SwiftUI, – url:https://developer.apple.com/documentation/swiftui/,

Що таке UI/UX/ - url: https://habr.com/ru/articles/321312/.
5. Документація Firebase. – url:https://firebase.google.com/docs?hl=ru.
6. Діаграма використання - url:https://habr.com/ru/articles/566218/.
7. Діаграма діяльності. – url:https://www.guru99.com/uk/uml-activity-diagram.html.
8. Концептуальна модель. url: https://www.guru99.com/uk/data-modelling-conceptual-logical.html.
9. Діаграмма послідовності - url: https://ru.wikipedia.org/wiki/Діаграмма_послідовності.
10. Діаграмма компонентів - https://creately.com/?aff=takeads.
11. Технічний проект по програмі - url:https://swrit.ru/tehproekt-na-po.
12. Функціональне тестування - url:https://browserstack.com/guide/functional-testing.
13. Guide Line по Swift – url: https://github.com/ILYA2606/SwiftGuideline.

Додаток А

Технічне завдання до додатку «Pizza Shop»

Вступ

Розроблюваний застосунок, що має робочу назву «Pizza Shop», призначена для замовлення їжі користувачем.

1 Підстави для розробки

Підставою для розробки програмного забезпечення є завдання до виконання кваліфікаційної роботи за спеціальністю «Інженерія програмного забезпечення».

Робота повинна бути виконана відповідно до технічного завдання, а замовник повинен отримати:

- Вихідний код додатку
- Опис додатку
- Інструкцію користувача.

2 Призначення розробки

2.1 Функціональне призначення

Розроблений програмний продукт має виконати дію отримання замовлень.

2.2 Експлуатаційне призначення

Розроблений програмний продукт має слугувати: для бізнесу - це прискорення роботи закладу та збільшенню обсягів продажів, для користувача – економія часу.

Вимоги до додатку

3.1 Вимоги до функціональних характеристик

3.1.1 Вимоги до складу виконуваних функцій

а) «Авторизація»:

- Створення облікового запису:

  Вхідна інформація: введення пошти, паролю, та підтвердження паролю.

  Вихідна інформація: завантаження даних входу облікового запису до бази.
- Перевірка наявності облікового запису:

  Вхідна інформація: пошта користувача та пароль.

  Вихідна інформація: перевірка пошти та паролю.
- Вхід до облікового запису:

  Вхідна інформація: Відсутня.

  Вихідна інформація: завантаження даних користувача.

б) «Колекція»:

- Відображення колекції:

  Вхідна інформація: код відображення колекцій.

  Вихідна інформація: отримання даних колекції.
- Натискання на колекцію:

  Вхідна інформація: код колекції.

  Вихідна інформація: переміщення до продукції.
- Переміщення до продукції:

  Вхідна інформація: отримання опису з бази.

  Вихідна інформація: Опис продукції.
- Вибір розміру та кількості продукції:

  Вхідна інформація: поточна сума продукції.

  Вихідна інформація: загальна сума.
- Додавання до кошика:

  Вхідна інформація: загальна сума, кількість позицій.

  Вихідна інформація: відправлення до кошика.

в) «Кошик»:

- Відображення поточних замовлень:

  Вхідна інформація: Відсутня.

  Вихідна інформація: відображення вибраної продукції та ціни.

- Замовлення піци:

Вхідна інформація: Загальна сума.

Вихідна інформація: Відправлення замовлення до адміністратора та відображення вибраного замовлення у профілі користувача.

г) «Обліковий запис»

- Відображення профілю користувача:

Вхідна інформація: отримання даних користувача.

Вихідна інформація: перегляд поточного замовлення та стадій його виконання.

- Вихід з облікового запису:

Вхідна інформація: поточний обліковий запис.

Вихідна інформація: вихід до «Авторизації».

д) Обліковий запис Адміністратора:

- Вхід в обліковий запис Адміністратора:

Вхідна інформація: пошта, пароль.

Вихідна інформація: перевірка даних входу та унікального ключа у базі.

- Відображення поточних замовлень:

Вхідна інформація: отримання даних.

Вихідна інформація: дані про користувача, його замовлення та стадії замовлення.

- Перевірка позицій замовлення:

Вхідна інформація: отримання з бази.

Вихідна інформація: відсутня.

- Додавання нових продукцій:

3.1.2 Вимоги до розміру пам’яті, необхідної для роботи додатку

Вимоги до об’єму затрачуваної оперативної пам’яті виділити неможливо, тому що об’єм оперативної пам’яті залежить від кількості товарів у додатку.

Дисковий простір, необхідний для збереження додатку неповинен перевищувати 15МВ.

3.2 Вимоги до надійності

3.2.1 Вимоги до надійного функціонування

Програма повинна нормально функціонувати при безперебійній роботі телефона. При виникненні збою в роботі апаратури, ресурси додатку не повинні бути пошкоджені, а відновлення роботи має проводитись після:

- перезавантаження операційної системи;
- перепідключення до інтернету.

3.2.2 Вимоги до контролю вхідної і вихідної інформації

Система повинна забезпечувати правильне введення інформації за рахунок використання, там де це доцільно, шаблонів введення, процедурного блокування введення некоректної інформації, списків та авто підстановки.

3.2.3 Вимоги до часу відновлення після відмови

Час відновлення після відмови, не пов’язаної з роботою додатку, повинен складатися з: часу перезапуску операційної системи; часу повторного зчитування з носіїв даних.

3.2.4 Вимоги до умов експлуатації та зберігання

Додаток не потребує спеціалізованих навичок для користування. Інтерфейс користувача зручний та інтуїтивно зрозумілий. Додаток призначена для експлуатації людьми різного віку та статі.

3.2.5 Вимоги до складу й параметрів технічних засобів

Для запуску додатку потрібен телефон або планшет зі системними характеристиками, котрі не будуть нижчі за наступні:

- процесор A6 компанії Apple;
- оперативна пам'ять з об'ємом 1GB;
- не менше 15MB вільного внутрішнього простору.

4 Вимоги до програмної документації

Документація до програмного засобу повинна містити наступні документи:

- технічне завдання;
- опис програми;
- текст програми;
- програма і методика випробувань;
- інструкція користувача.

5 Стадії та етапи розробки

Стадії та етапи розробки гри та терміни їх виконання повинні відповідати затвердженому графіку проектування кваліфікаційної роботи та зводитись до таких, що наведені в таблиці А.1.

Таблиця А.1– Стадії, етапи розробки програми, та терміни їх виконання

Стадії розробки	Етапи робіт	Зміст робіт	Терміни виконання
1	2	3	4
Технічне завдання	Розробка технічного завдання.	Обґрунтування необхідності розробки програми	05.03.24 – 26.03.24
	Затвердження технічного завдання.	Розробка пояснювальної записки. Узгодження і затвердження технічного завдання.	26.03.24 – 06.04.24
Ескізний проект	Розробка ескізного проекту	Попередня розробка структури вхідних та вихідних даних, уточнення методів рішення завдань, загальний опис алгоритму.	06.04.24 – 09.04.24

Кінець таблиці А.1

1	2	3	4
	Затвердження ескізного проекту	Розробка пояснювальної записки. Узгодження і затвердження ескізного проекту.	09.04.24 – 13.04.24
Технічний проект	Розробка технічного проекту	Уточнення структури вхідних та вихідних даних, розробка алгоритму рішення задачі, розробка структури програми.	13.04.24 – 17.04.24
	Затвердження технічного проекту	Розробка пояснювальної записки. Узгодження і затвердження технічного проекту.	17.04.24 – 20.04.24
Робочий проект	Розробка програми	Програмування та налагодження програми	20.04.24 – 27.04.24
	Випробування програми	Розробити, узгодити та затвердити програму і методику випробувань, коригування програми за результатами випробувань.	27.04.24 – 05.05.24
	Розробка програмної документації	Розробка програмної документації	05.05.24 – 11.05.24

6. Порядок приймання та контролю

Для контролю й приймання необхідно надати інструкцію користувача та опис програми, а також програму та методику випробування.

Порядок контролю і приймання даної розробки здійснюється представником замовника у присутності представників розробника згідно з програмою та методикою випробувань.

За результатами приймання складається акт, який підписується представником замовника та представником розробника та утверджується керівниками організації-замовника і організації-розробника.

У випадку виявлення помилок під час приймання програмного виробу складається акт про виявлені помилки, який підписується представниками замовника та розробника й стверджується керівниками організації -замовника й організації-розробника. Розробник повинен в продовж не більш як 1-го місяця видалити вказані зауваження, й повідомити замовника про повторне проведення перевірки, не пізніше як за 2 тижні.

Додаток Б
Програма і методика випробувань додатку «Pizza Shop»

1 Об’єкт випробування

Об’єктом випробування є «Pizza Shop», котра призначена для прискорення та збільшенню об’ємів продажів.

2 Мета випробування

Метою випробування є перевірка додатку на наявність помилок і відповідність розробленого програмного забезпечення вимогам, які зазначені у технічному завданні у додатку А. Якщо прийнятий рівень відповідності додатку вимогам досягнутий, а також відсутні помилки, то програмний продукт приймається в експлуатацію.

3 Вимоги до програми

При проведенні випробувань функціональні характеристики додатку підлягають перевірці на відповідність вимогам, викладеним у пункті «Вимоги до функціональних характеристик» технічного завдання.

Для проведення випробування будемо розглядати такі функції програмного продукту:

- Функція створення облікового запису;
- Функція вибору замовлення;
- Функція відправки замовлення до Адміністратора;
- Функція збереження даних користувача та його замовлення.

4 Вимоги до програмної документації

Документація, що пропонується до випробування програмного забезпечення для автоматизації обліку трудових ресурсів підприємства, повинна включати наступні документи:

- технічне завдання;

- опис програмного забезпечення;
- текст програмного забезпечення;
- програму та методику випробувань;
- інструкцію користувача.

5 Засоби та порядок випробувань

Засоби випробувань

Для технічного та програмного випробування даного застосунку необхідно, телефон або планшет компанії Apple з операційною системою IOS 10.6 та вище.

Порядок проведення випробувань

Випробування додатку буде проводитися протягом кількох днів. Розробник самостійно проводить тестування усіх функцій програми у присутності інших учасників проекту, якщо такі є. При наявності відповідної можливості у процесі проведення випробувань можуть додатково брати участь інші незалежні але зацікавлені особи. Попередньо до початку випробувань визначається кінцевий склад групи, що буде проводити випробування. Кожен член групи одержує необхідну документацію до програми.

При випробуванні проводиться:

- контроль складу виконуваних функцій;
- контроль коректності виконання окремих функцій;
- випробування на правильність роботи окремих модулів;
- комплексне випробування.

При виявленні помилок у роботі програмного продукту розробник складає перелік цих помилок і самостійно визначає термін їхнього виправлення. Після цього розробник проводить повторне випробування у повному обсязі.

6 Методи випробування

Тестування буде проводитися функціональне.

Програма випробування:

1) Випробування функції «Авторизації»:

Відображаються текстові поля та кнопка входу та кнопка створення облікового запису.

- Коли введено дані нового користувача у текстові поля Авторизації, натиснуто кнопку «Вхід», відкривається повідомлення «користувач не існує».
- Якщо обліковий запис вже створений, відбувається перевірка наявності облікового запису.

2) Випробування функції «Створення облікового запису»:
   - Коли натиснуто кнопку «Створення облікового запису» екран розширюється та додається додаткове текстове поле для введення даних.
   - Обліковий запис додається до бази
   - При не коректно введених даних відкривається повідомлення, що потрібно перевірити правильність.
3) Випробування функції отримання товарів.
   - При вході у обліковий запис, відкривається «Каталог» який отримав дані з бази даних, у даному випадку це назви позицій, ціни, описи, та картинки.
4) Випробування функції вибору замовлень.
   - Після відкриття «Каталогу» було обрано замовлення, його кількість та розмір.
   - При додані до корзини, коректно відображається загальна сума замовлення.
5) Випробування функції відображення даних користувача.
   - Після вибору замовлення та його підтвердження, у профілі користувача відображаються його поточні замовлення.
   - При введені даних користувача, дані зберігаються.
   - Також при повторному вході дані користувача відображаються коректно.
6) Випробування функції виходу з облікового запису.

- Після натискання кнопки виходу, відкривається вікно підтвердження, в якому є два варіанта:
- здійснити вихід,
- відмінити.
- Якщо обрати «відмінити», вікно закривається.
- Якщо обрати «здійснити вихід», відбувається повернення до Авторизації.

7) Випробування колекції замовлень Адміністратора.
   - Після входу у обліковий запис адміністратора, відображається колекція поточні замовлення користувачів, які завантажуються з бази даних.

Також присутні функції:

- перегляд позицій,
- оновлення,
- виходу.

- При натисканні на позицію, відкривається інформаційне вікно про з детальним описом поточного замовлення, також можливо змінити статус замовлення.
- При натисканні «Оновлення», здійснюється оновлення статусу замовлення.
- При натисканні «Виходу», відбувається повернення до Авторизації.

8) Випробування функції додавання позиції.

   При натисканні кнопки «Add Order» у колекції Адміністратора, відкривається вікно з текстовими полями:

   - назва,
   - опис,
   - ціна.

   - Після введення відбувається перевірка всіх заповнених даних, при відсутності даних система очікує.
   - Коли всі дані введені, інформація зберігається.

Додаток В

Інструкція користувача до додатку «Pizza Shop»

1 Опис додатку

Додаток включає в себе: продукцію, обліковий запис, інформацію користувача та базу даних.

- Продукція – це колекція елементів, в яку можливо додавати нову продукцію та обирати її.
- Обліковий запис – це сутність користувача, яка дозволяє виконувати вхід в додаток та дає йому унікальний ключ в базі даних для подальшого використання додатку. Також при бажані залишатися у додатку при закриті.
- Інформація про користувача – це дані користувача, які по унікальному ключу зберігаються у базі даних його: ім’я, номер телефону, адреса та його замовлення.
- База даних додатку - це організована структура, яка призначена для зберігання, зміни та обробки взаємозалежної інформації.

  Які структури даних включає в себе:

  - Замовлення(Orders).
  - Продукції(Products).
  - Користувачів(Users).
- Замовлення(Orders) – це структура даних, яка включає в себе інформацію про поточні замовлення: користувачів які замовили, час коли було замовлено та етапи виконання замовлення.
- Продукції(Products) – це структура даних, яка містить у собі інформацію про поточні товари: опис, назву, ціну. Саме сюди додаються нові товари.
- Користувач(User) – це структура даних, яка містить інформацію про користувача: адресу, ім’я, номер телефону та їх унікальний ключ. Сюди додаються нові користувачі.

2 Інструкція.

Перед входом у додаток, користувач повинен ввести пошту та пароль.

У разі, якщо користувач вирішить створити обліковий запис, то він має заповнити 3 текстові поля: пошту, пароль та підтвердження паролю. Після введення даних відкриється вікно, зображене на рисунку В.2.

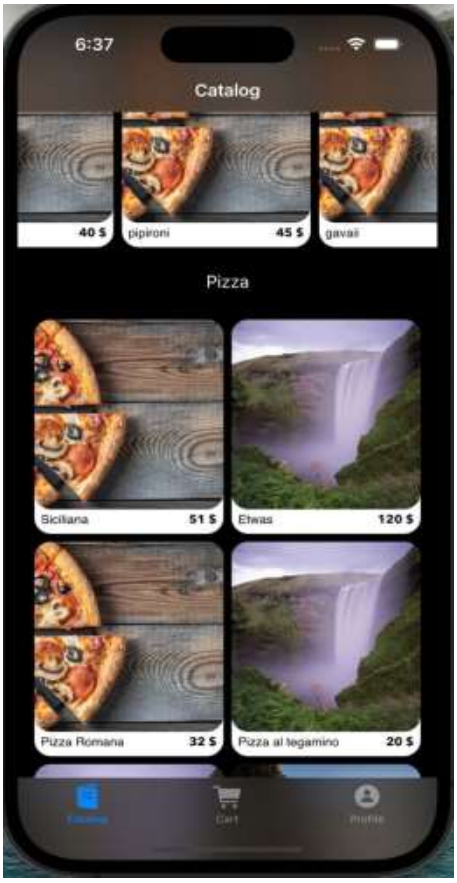

Рисунок В.2 – Колекція Товарів

Після вибору товару, відкривається вікно продукції, в якому також потрібно вибрати кількість та розмір. По натисканню кнопки «Add to Trash», замовлення відправляється у «Корзину»

На рисунку В.3 зображене вікно Продукції.

Коли замовлення обрано потрібно перейти до корзини та підтвердити замовлення. Після воно відправляється до бази та відображається у Профілі користувача. На рисунку В.4 зображено обліковий запис користувача.

Рисунок В.3 – Інтерфейс вікна Продукції

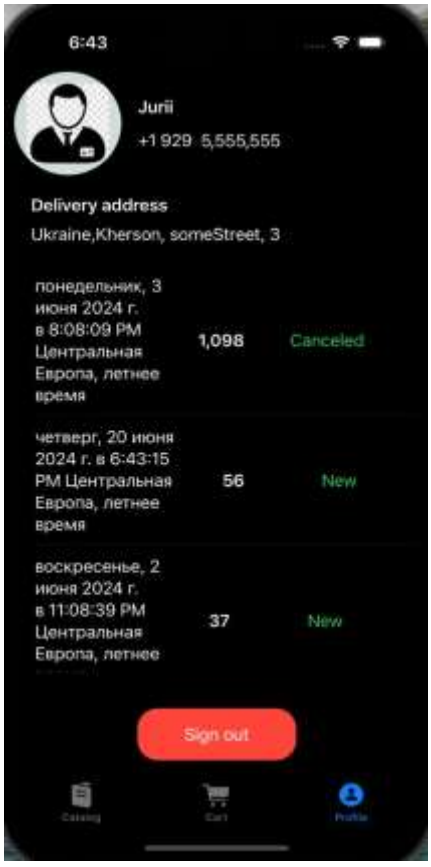

Рисунок В.4 – Обліковий запис користувача

Інтерфейс Адміністратора.

Після входу у обліковий запис адміністратора, відображається колекція поточних замовлень користувачів, функції:

- додавання,
- оновлення,
- зміни статусу та перегляд позицій.

На рисунку В.5 зображено інтерфейс Адміністратора.

Рисунку В.5 – Зображення інтерфейсу Адміністратора

Окрім головного інтерфейсу, адміністратору також доступна можливість перегляду позицій та зміни статусу замовлення. Зображено на рисунку В.6.

Рисунок В.6 – Інтерфейс вікна з відображенням рахунку

Функція додавання позицій.

По натисканню кнопки «Add Order», відкривається вікно створення позиції як зображено на рисунку В.7

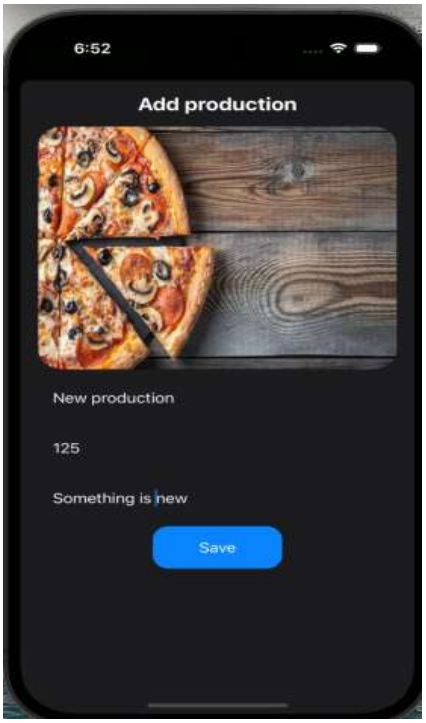

Рисунок В.7 – Вікно додавання нової позиції

Додаток Г

Опис додатку «Pizza Shop»

1 Загальні відомості

Назва: «Pizza Shop».

Область застосування: їжа та напої.

2 Функціональне призначення

Функціональне призначення розробки: додаток «Pizza Shop» має слугувати інструментом для прискорення та збільшенню об’ємів продажів.

3. Технічні засоби

Для запуску потрібен телефон або планшет зі системними характеристиками, котрі не будуть нижчі за наступні:

- процесор A6 компанії Apple;
- оперативна пам'ять 1GB;
- не менше 15MB вільного дискового простору.

Додаток Д
Текст виконуючого коду додатку «Pizza Shop»

Текст виконуючого коду додатку «Pizza Shop» (фрагмент)

```
import SwiftUI
import Firebase
import FirebaseAuth
let screen = UIScreen.main.bounds
@main
struct PizzaShopApp: App {
    @UIApplicationDelegateAdaptor private var appDelegate: AppDelegate
    var body: some Scene {
     WindowGroup {
        if let user = AuthService.shared.currentUser {
           if user.uid == "r4gz8Wf8RMQn8uv3FHVDutG9paI3" {
             AdminOrdersView()
           } else {

              let viewModel = MainTabBarViewModel(user: user)
              MainTabBar(viewModel: viewModel)
           }
        } else {
           AuthView()
        }
     }
  }
     class AppDelegate: NSObject, UIApplicationDelegate {
     func application(_ application: UIApplication, didFinishLaunchingWithOptions
launchOptions: [UIApplication.LaunchOptionsKey : Any]? = nil) -> Bool {
```

```
            FirebaseApp.configure()
            print("ok")
            return true
        }
    }
}
import Foundation
import FirebaseFirestore
struct Order {
var id: String = UUID().uuidString
    var userID: String
    var positions = [Position]()
    var date: Date
    var status: String
        var cost: Int {
                var sum = 0
        for position in positions {
            sum += position.cost
        }
        return sum
    }
        var representation: [String: Any] {
        var repres = [String: Any]()
        repres["userID"] = userID
        repres["date"] = Timestamp(date: date)
        repres["id"] = id
        repres["status"] = status
        return repres
    }
```

```
        init(id: String = UUID().uuidString,
         userID: String,
         positions: [Position] = [Position](),
         date: Date,
         status: String) {
        self.id = id
        self.userID = userID
        self.positions = positions
        self.date = date
        self.status = status
    }
        init?(doc: QueryDocumentSnapshot) {
        let data = doc.data()
                guard let id = data["id"] as? String else { return nil }
        guard let userID = data["userID"] as? String else { return nil }
        guard let date = data["date"] as? Timestamp else { return nil }
        guard let status = data["status"] as? String else { return nil }
                self.id = id
        self.userID = userID
        self.date = date.dateValue()
        self.status = status
        }
}
import Foundation
enum OrderStatus: String, CaseIterable {
    case new = "New"
    case cooking = "Cooking"
    case delivery = "Delivery"
    case completed = "Completed"
```

```
    case cancel = "Canceled"
}
import Foundation
import FirebaseFirestore
struct Position: Identifiable {
   var id: String
    var product: Product
    var count: Int
        var cost: Int {
        return product.price * self.count
    }
        var representation: [String: Any] {
        var repres = [String: Any]()
        repres["id"] = id
        repres["count"] = count
        repres["title"] = product.title
        repres["price"] = product.price
        repres["cost"] = cost
        return repres
    }
        internal init(id: String, product: Product, count: Int) {
        self.id = id
        self.product = product
        self.count = count
    }
        init?(doc: QueryDocumentSnapshot) {
        let data = doc.data()

        guard let id = data["id"] as? String else { return nil }
```

```
        guard let title = data["title"] as? String else { return nil }
        guard let price = data["price"] as? Int else { return nil }
        let product: Product = Product(id: "",
                                       title: title,
                                       imageUrl: "",
                                       price: price,
                                       descript: "")
        guard let count = data["count"] as? Int else { return nil }
              self.id = id
        self.product = product
        self.count = count
    }
}
import Foundation
import FirebaseFirestore
struct Product {
  var id: String
    var title: String
    var imageUrl: String = ""
    var price: Int
    var descript: String
    var representation: [String: Any] {
        var repres = [String: Any]()
        repres["id"] = self.id
        repres["title"] = self.title
        repres["price"] = self.price
        repres["descript"] = self.descript
        return repres
    }
```

```
        internal init(id: String = UUID().uuidString,
                title: String,
                imageUrl: String = "",
                price: Int,
                descript: String) {
        self.id = id
        self.title = title
        self.imageUrl = imageUrl
        self.price = price
        self.descript = descript
    }
        init?(doc: QueryDocumentSnapshot) {
        let data = doc.data()
        guard let id = data["id"] as? String else { return nil }
        guard let title = data["title"] as? String else { return nil }
        guard let price = data["price"] as? Int else { return nil }
        guard let descript = data["descript"] as? String else { return nil }
        self.id = id
        self.title = title
        self.price = price
        self.descript = descript
    }
}
import Foundation
struct RNBUser: Identifiable {
     var id: String = UUID().uuidString
    var name: String
    var phone: Int
    var address: String
```

```
    var representation: [String: Any] {
        var repres = [String: Any]()
        repres["id"] = self.id
        repres["name"] = self.name
        repres["phone"] = self.phone
        repres["address"] = self.address
        return repres
    }
}
import Foundation
import FirebaseAuth
class AuthService {
       static let shared = AuthService()
        private init() { }
       let auth = Auth.auth()
  var currentUser: User? {
        return auth.currentUser
    }
        func signOut() {
        try! auth.signOut()
    }
       func signUp(email: String,
                password: String,
                completion: @escaping (Result<User, Error>) -> () ) {
              auth.createUser(withEmail: email, password: password) { result, error in
            if let result = result {
                let rnbUser = RNBUser(id: result.user.uid,
                                      name: "",
```

```
                    phone: 0,
                    address: "")
                DatabaseService.shared.setProfile(user: rnbUser) { resultDB in
            switch resultDB {
            case .success(_):
                completion(.success(result.user))

            case .failure(let error):
                completion(.failure(error))
            }
        }
    } else if let error = error {
        completion(.failure(error))
    }
  }
}
```