**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**Сумський державний університет**
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук

«До захисту допущено»
В.о. завідувача кафедри
_____ Ігор ШЕЛЕХОВ
(підпис)
червня 2023 р.

# КВАЛІФІКАЦІЙНА РОБОТА
## на здобуття освітнього ступеня бакалавр

зі спеціальності 122 - Комп'ютерних наук,
освітньо-професійної програми «Інформатика»
на тему: «Інформаційне та програмне забезпечення клієнтської частини мобільного застосунку для ресторанного бізнесу»
здобувачки групи ІН - 91 Гаркавенко Аліни Євгеніївни

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

_____ Аліна Гаркавенко
(підпис)

Керівник, асистент, кандидат
фізико-математичних наук Ольга Шутилєва _____
(підпис)

**Суми – 2023**

# АНОТАЦІЯ

**Записка:** пояснювальна записка складається зі вступу, розділів, висновків, списку використаних джерел із 18 найменувань, трьох додатків. Загальний обсяг роботи складає 95 сторінок, у тому числі 49 сторінки основного тексту, 2 сторінки списку використаних джерел, 44 сторінок додатків.

**Об’єкт дослідження:** інформаційне та програмне забезпечення клієнтської частини мобільного застосунку для ресторанного бізнесу.

**Мета роботи:** надання можливості користувачу на основі отриманої інформації виконувати замовлення страв та переглядати весь наданий асортимент ресторану.

**Методи дослідження:** аналіз аналогів мобільних додатків, аналіз інструментів для реалізації поставленої задачі.

**Результат:** розроблена клієнтська частина мобільного застосунку для ресторанного бізнесу.

**Ключові слова:** мобільний додаток, ReactJS, NodeJS, react navigation, redux\saga.

# ЗМІСТ

# ВСТУП

Хоча люди все ще можуть користуватися інтернетом зі стаціонарного комп'ютера, мобільні додатки стають все більш популярним способом взаємодії в мережі.

Не кожному малому та середньому бізнесу потрібен мобільний додаток, але в правильній ситуації він може бути корисним. Мобільні додатки можуть допомогти формувати лояльність до бренду та контролювати ділові відносини, особливо якщо вони підтримують тісний контакт з клієнтами або діловими партнерами. Вони також можуть посилити рекламні зусилля, покращити аналітику клієнтів та полегшити онлайн-покупки. Керівники підприємств повинні вивчити переваги мобільних додатків, щоб визначити, чи може додаток допомогти їхньому бізнесу розвиватися.

У цифровому світі клієнти вимагають гнучкості при виборі та замовленні продуктів і послуг. Організації можуть розробляти мобільні додатки для інформування та здійснення покупок через додаток.

Компанії почали дозволяти клієнтам робити замовлення через мобільні додатки на початку 2010-х років. До цього часу клієнти не наважувалися розміщувати замовлення на товари, особливо дорогі - в додатках. Однак, коли користувачам стало зручніше здійснювати покупки зі своїх смартфонів, багато з них почали надавати перевагу цьому каналу. Згідно зі звітом Statista, з третього кварталу 2021 року по третій квартал 2022 року замовлення товарів через мобільні пристрої зросли більш ніж на 20%. [1]

На основі аналізу аналогів  було визначено мету кваліфікаційної роботи бакалавра: розробка інформаційного та програмного забезпечення клієнтської частини мобільного застосунку для ресторанного бізнесу з використанням фреймворку ReactNative.

Для досягнення мети потрібно вирішити наступні задачі:

- виконати аналіз предметної області;

- розробити структуру клієнтської частини;
- розробити дизайн клієнтської частини;
- розробити модулі оформлення та оплати замовлення;
- провести тестування додатку.

Практичне значення інформаційного та програмного забезпечення клієнтської частини мобільного застосунку для ресторанного бізнесу полягає у наданні можливості користувачу на основі отриманої інформації виконувати замовлення страв та переглядати весь наданий асортимент ресторану.

# 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

## 1.1 Актуальність проблеми

Кожен бізнес прагне оптимізувати свої процеси та досягати поставлених цілей ефективніше, швидше та дешевше. Досягнувши цих цілей, компанія отримує переваги над конкурентами та ресурси для подальшого розвитку.

Для досягнення своїх цілей компанії вдаються до різноманітних інструментів, включаючи веб-сайти, CRM/ERP-системи, хмарні сервіси та мобільні додатки. Однак додатки все ще використовуються рідше порівняно з іншими, хоча вони є найефективнішим ресурсом для досягнення деяких завдань.

Кожен бізнес хоче розвиватися на цифровій платформі, а боротьба за увагу клієнтів та покращення користувацького досвіду робить кожну галузь по-справжньому конкурентоспроможною. Щоб привернути увагу мобільних користувачів, потрібен зручний додаток, який може вирішити проблеми користувачів, розважити та полегшити їхнє повсякденне життя.

Ідея додатку має бути однаковою для всіх пристроїв. Розробник повинен забезпечити значну функціональну ефективність. Крім того, функції повинні бути сумісними з усіма пристроями та платформами. Це збільшує охоплення додатку і допомагає йому стати найбільш використовуваним. [2]

Якщо додаток не відповідає всім цим критеріям, він може не мати успіху. Щоб уникнути таких проблем, вам подбати про розміри, роздільну здатність та пікселі кожної версії додатку на різних мобільних пристроях.

Перед тим як створити мобільний додаток, потрібно проаналізувати:

- актуальність інстументів для розробки: фреймворки, бази даних, мови програмування:
- додатки-аналоги для підтримки ресторанного бізнесу;
- працеспроможність інструментарію, який буде використаний для розробки;

- тренди UX/UI;

Ринок мобільних додатків сьогодні дуже конкурентний. Тому дуже важко створити додаток, який би вирізнявся з-поміж інших. Просування додатку - одна з найбільших проблем при розробці мобільних додатків. І на даний момент існує багато способів зробити це.

Зробити так, щоб його помітили і знайшли - один із цих способів. Якщо користувачі можуть легко знайти його на ринку додатків, вони зможуть його завантажити. А якщо додаток є високофункціональним і пропонує дивовижний користувацький досвід, але його нелегко знайти, то неможливо досягти поставлених бізнес-цілей. [3]

Тому мобільний додаток це не лише зручність для цільової аудиторії, а також економія ресурсів завдяки функціональним можливостям додатку.

## 1.2 Аналіз існуючих аналогів

Ресторанні технології зробили революцію в управлінні ресторанами, замінивши важкий і виснажливий рутинний менеджмент на ефективність, автоматизацію та чіткість.

Найкращі додатки для управління рестораном дозволяють керувати бізнесом з будь-якого місця, надають цінну інформацію, розширюють можливості персоналу і дозволяють зосередитися на створенні найкращого досвіду для користувачів.

У 2023 році залишиться не так багато сфер в управлінні рестораном, які неможливо покращити за допомогою технологій. Програмне забезпечення для бронювання допомагає організувати роботу персоналу, сайти з відгуками надають дієвий зворотній зв'язок про роботу закладу, а програмне забезпечення для управління персоналом синхронізує роботу команди з автоматичним плануванням.

**TouchBistro** [4]

Першим додатком для підтримки ресторанного бізнесу є TouchBistro. Минули ті часи, коли доводилося затримуватися біля входу до ресторану, щоб заплатити за їжу. TouchBistro - додаток для торгових точок для iPad - дозволяє офіціантам приймати платежі прямо за столом, менеджерам контролювати активність продажів, а бухгалтерам та адміністраторам отримувати швидкий доступ до цінних тенденцій продуктивності. Вражаючий перелік функцій також включає можливість вносити зміни до меню та інвентарю в режимі реального часу, а також з легкістю розділяти рахунки - завдання, яке часто спричиняє головний біль.

На рисунку 1.1 представлено вигляд додатку TouchBistro.

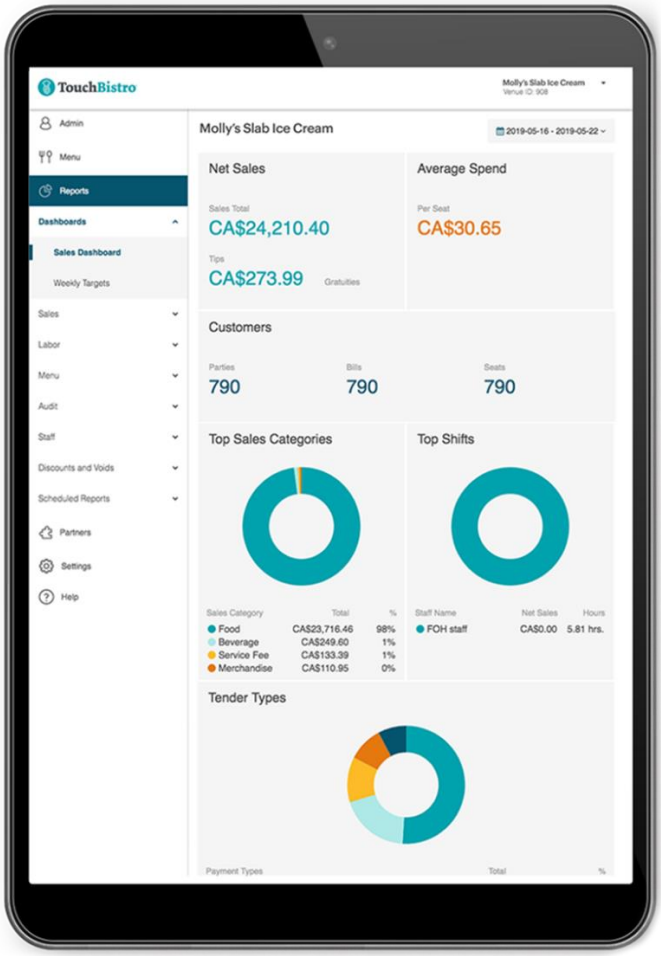


Рисунок 1.1 – вигляд додатку TouchBistro

**Zuppler** [5]

Zuppler - це перспективний мобільний додаток для ресторанних замовлень, який дозволяє власникам ресторанів створювати кастомізовані меню онлайн-замовлень для відвідувачів, які працюють з додатком через головну панель.

Окрім інтуїтивно зрозумілого меню та безлічі опцій для роботи, ця платформа також відома тим, що допомагає операторам брендових ресторанів створювати акційні пропозиції, пропозиції зі знижками та інші сезонні спеціальні пропозиції, які можна розміщувати на додаток до меню замовлення їжі. Ці налаштування можуть бути додані на сайт вашого ресторану і на мобільну платформу замовлення їжі. На рисунку 1.2 зображено інтерфейс додатку Zuppler.

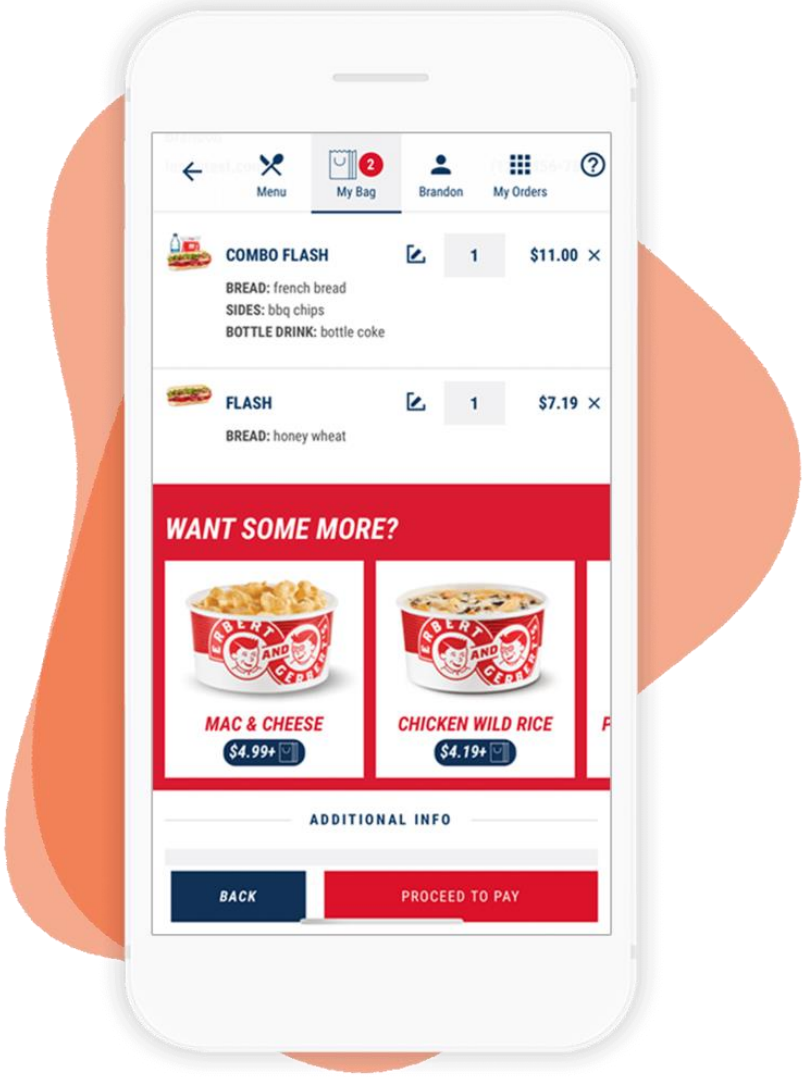


Рисунок 1.2 – вигляд додатку Zuppler

**Toast POS** [6]

Це надійний додаток, поєднаний з повноцінним середовищем для управління рестораном, де є достатньо інструментів для виконання замовлень, управління взаємовідносинами з клієнтами (CRM), бухгалтерського обліку, аналітики та інших функцій, необхідних для успішного ведення ресторанного бізнесу.

Сповіщення Toast POS надсилаються на мобільні пристрої або в POS-систему для зручного управління замовленнями.

Щоб отримати найкращий досвід роботи з Toast POS як спеціальним додатком для замовлення їжі для ресторанів, потрібно отримати доступ до модуля

CRM, щоб попередньо продавати щось своїм клієнтам. Попередній продаж включає в себе розробку меню, роздачу їжі або знижки і може зайняти деякий час, щоб показати результати.

На рисунку 1.3 зображено інтерфейс додатку Toast POS.

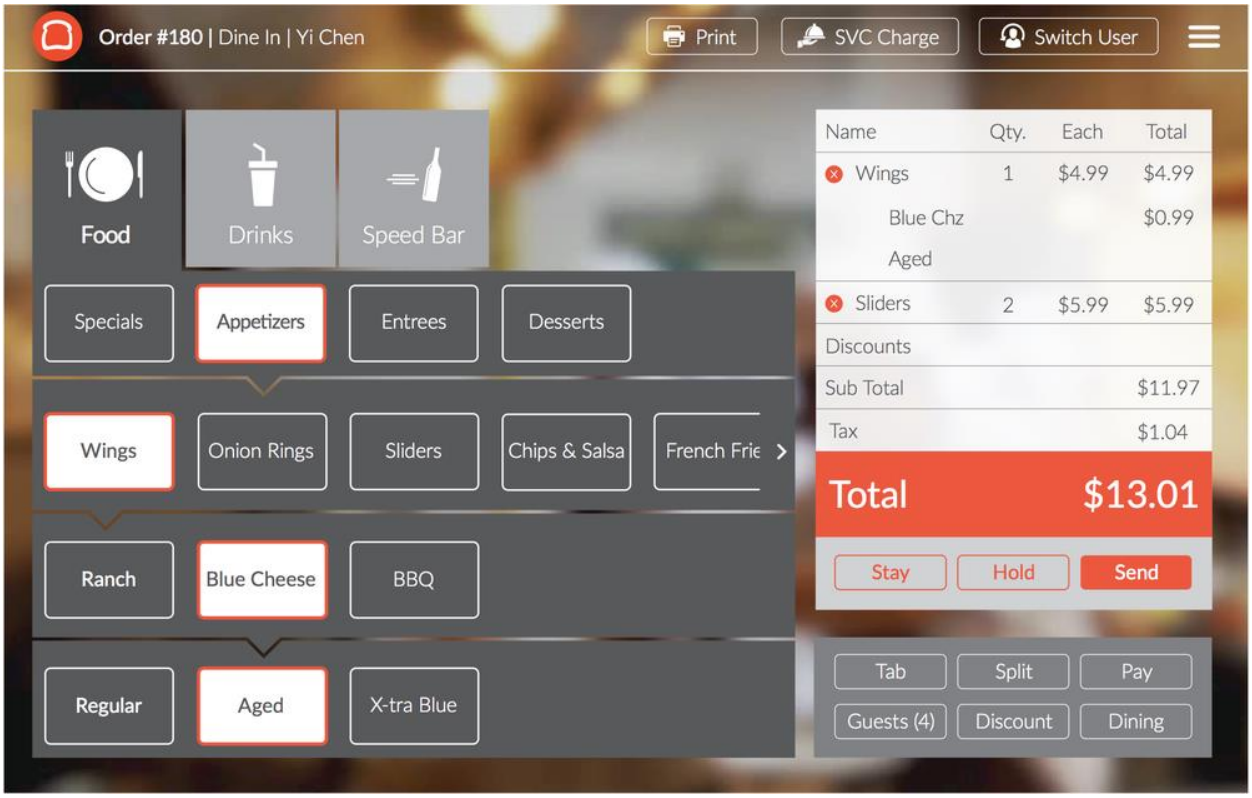


Рисунок 1.2 – вигляд додатку Toast POS

Згідно аналізу аналогів, була створена порівняльна таблиця аналогів та прикладів мобільних додатків для підтримки ресторанного бізнесу. Результати аналізу представлено у таблиці 1.1.

Таблиця 1.1 – Порівняльна таблиця аналогів

| Критерії порівняння | Додатки - аналоги | | |
|---|---|---|---|
| | TouchBistro | Zuppler | Toast POS |
| Сучасний дизайн | + | + | + |
| Доступ без реєстрації | - | - | + |
| Зручність навігації | + | + | + |

Продовження таблиці 1.1 – Порівняльна таблиця аналогів

| | | | |
|---|---|---|---|
| Кросс-платформна підтримка | - | - | + |
| Присутність зворотнього зв’язку | + | + | + |
| Наявність інструкції користувача | - | + | + |

## 1.3 Постановка задачі

Метою кваліфікаційної роботи бакалавра є інформаційне та програмне забезпечення клієнтської частини мобільного застосунку для ресторанного бізнесу.з використанням фреймворку ReactJS. Головною задачею мобільного додатку є надання можливості користувачу на основі отриманої інформації виконувати замовлення страв та переглядати весь наданий асортимент ресторану.

Мобільний додаток містить в собі наступні функції:

- пошук товару: по всім категоріям, по конкретній категорії;
- фільтрування товарів: за популярністю, по рейтингу, ціна за зростанням, за зменшенням, по ціновому діапазону;
- кошик;
- перегляд сповіщень;
- оформлення замовлення;
- методи зворотного зв'язку;
- меню налаштування інтерфейсу;

Дизайн мобільного додатку буде розроблений за останніми стандартами та трендами UX/UI, що дасть змогу конкурувати з аналогічними додатками. Сучасний підхід до реалізації користувацької системи повинен забезпечувати реалізацію подачі товару в стилі мінімалізму, а також повинен бути оптимізованим під всі доступні мобільні пристрої.

Для реалізації кваліфікаційної роботи бакалавра необхідно реалізувати наступне:

- провести аналіз технологій для реалізації мобільного додатку;
- розробити інтерфейс мобільного додатку;
- розробити функціонал клієнтської частини;
- інтегрувати функціонал з інтерфейсом додатку
- виконати тестування функціоналу мобільного додатку.

# 2 ІНСТРУМЕНТИ ДЛЯ ВИРІШЕННЯ ПОСТАВЛЕНИХ ЗАДАЧ

Вибір інструментів та технологій для розробки інформаційного та програмного забезпечення клієнтської частини мобільного застосунку для ресторанного бізнесу має велику важливість і безпосередній вплив на функціональність, продуктивність, ефективність та користувацький досвід.

Правильний вибір інструментів та технологій відіграє важливу роль у забезпеченні надійності та стабільності роботи мобільного застосунку, а також у забезпеченні оптимальної продуктивності та швидкості виконання. Наприклад, вибір відповідного фреймворку або платформи розробки може позитивно вплинути на ефективність роботи застосунку та сприяти швидкому відгуку на дії користувача.

Крім того, вибір відповідних інструментів та технологій повинен враховувати вимоги безпеки та захисту даних. Ресторанний бізнес залучає обробку та зберігання конфіденційної інформації, такої як особисті дані клієнтів або фінансові дані. Вибір надійних технологій для шифрування даних, захисту від несанкціонованого доступу та вразливостей може гарантувати конфіденційність та безпеку важливої інформації.

Процес розробки мобільного додатку включає наступні етапи:

- розробка шаблону екранів;
- програмування інтерактивних елементів;
- створення функцій замовлення товару;
- створення функцій для додавання обраних товарі до кошика та обраних;
- розробка інтуїтивно зрозумілого інтерфейсу з підказкками.

При визначенні необхідних елементів для реалізації мобільного додатку, було прийнято рішення про використання наступних технологій:

- React Native;
- NodeJS.

React Native - це крос-платформний фреймворк на основі JavaScript, який широко використовується для створення нативних мобільних додатків для платформ iOS та Android. Незважаючи на те, що React Native був запущений у 2015 році компанією Facebook, він швидко зайняв центральне місце як фреймворк з відкритим вихідним кодом. Зрештою, він набув популярності як найкраще рішення для розробки додатків після того, як на ньому було створено багато додатків, включаючи Skype, Instagram, Facebook та інші.

Завдяки перевагам багаторазового використання кодів, можливості швидкого тестування та запуску додатків як на iOS, так і на Android, React Native став ідеальною платформою для розробки додатків.

Ось декілька причин, чому React Native є найкращим вибором для розробки додатків.

**Заощаджує час і гроші**

Основна причина вибору React Native для розробки додатків - це економія часу, зусиль та коштів. Створення двох нативних додатків для різних платформ коштує дорожче, ніж розробка додатку з React Native на єдиній кодовій базі. Тож замість того, щоб наймати розробника мобільних додатків для створення двох нативних додатків, з React Native розробники можуть розгортати одне й те саме оновлення для платформ iOS та Android без зайвих зусиль, пов'язаних з використанням різних кодових баз.

**Висока продуктивність**

Продуктивність додатку завжди є головною проблемою для бізнесу. І додатки на React Native не поступаються нативним додаткам для Android та iOS. В той час як інші крос-платформні фреймворки використовують CPU, React Native використовує GPU для обчислювальних задач. Це основна причина, чому React Native додатки мають вищу продуктивність.

**Відкритий код**

React Native надає широкий вибір нативних інструментів інтерфейсу користувача для створення красивих додатків. Розробники не мають проблем з

міграцією React Native додатків з однієї платформи на іншу, оскільки React Native дозволяє розробникам легко експортувати коди з Android Studio до Xcode. [7]

## 2.1 Структурно-функціональне моделювання

Метод функціонального моделювання IDEF0 (англ. Integration Definition for Function Modeling) є формальним методом, що використовується для аналізу та опису функціональної структури системи. Цей метод базується на ієрархічній структурі, яка дозволяє розкрити взаємозв'язки між різними функціями та процесами в системі. IDEF0 забезпечує чіткий та систематичний підхід до моделювання функцій, використовуючи графічні символи та нотацію.

При створенні мобільного додатку метод функціонального моделювання IDEF0 може бути використаний для детального аналізу та уточнення функціональних вимог до додатку. Шляхом розбиття функціональності на більш прості елементи та визначення залежностей між ними, IDEF0 дозволяє отримати систематичне уявлення про те, які функції має виконувати додаток та як вони пов'язані між собою. Це дозволяє команді розробників мобільного додатку краще розуміти потреби користувачів та ефективно планувати його функціональні можливості. [8]

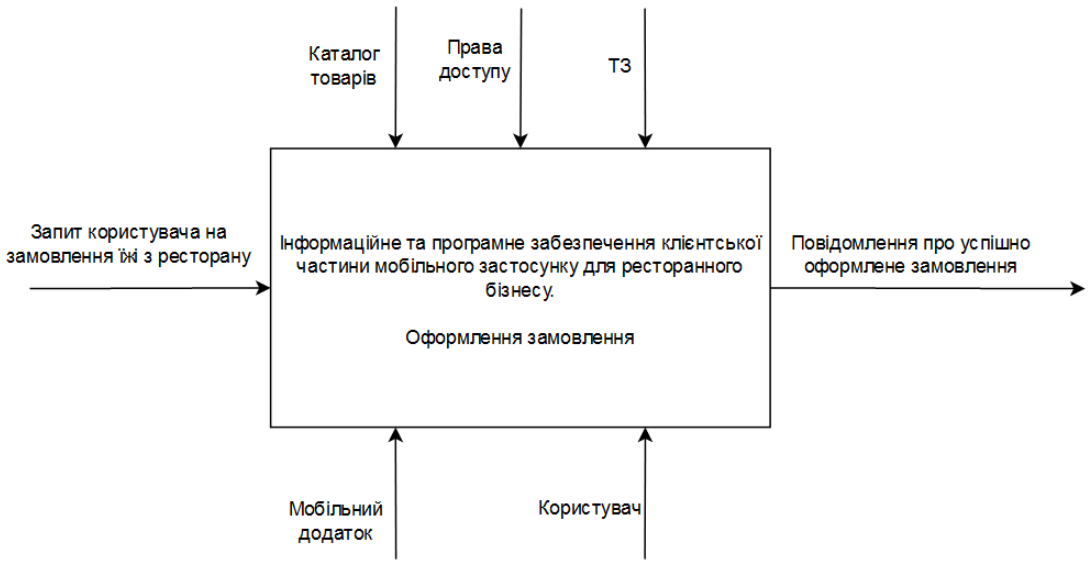


Рисунок 2.1 - Контекстна діаграма

В діаграмі визначені такі головні елементи, як:

- вхідні дані: запит користувача на замовлення їжі з ресторану;
- вихідні дані: повідомлення про успішно оформлене замовлення;
- управління: ТЗ (технічне завдання), каталог товарів та права доступу;
- механізми: мобільний додаток, користувач.

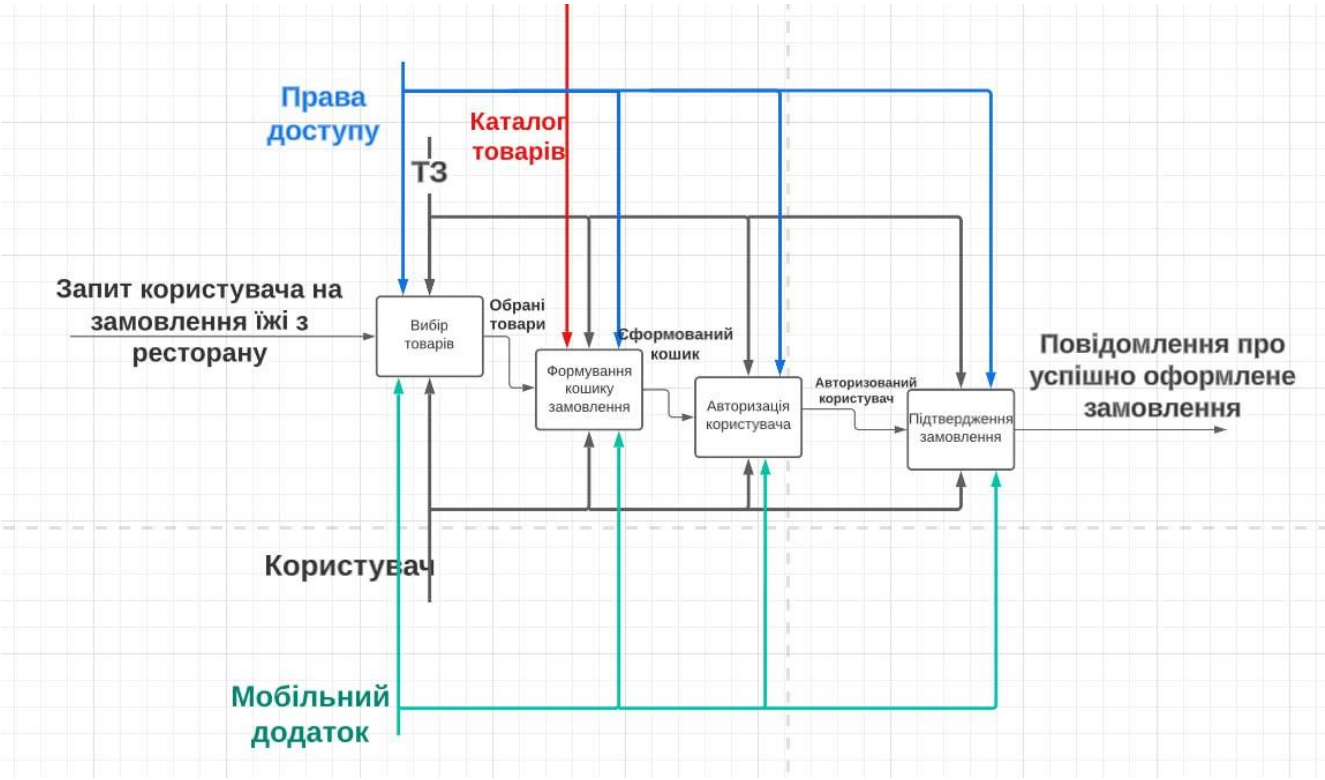


Рисунок 2.5 – Діаграма декомпозиції

Для відображення інформації про надані функції користування мобільного додатку для різних типів користувачів було створено діаграму варіантів використання [9] (рис 2.6).

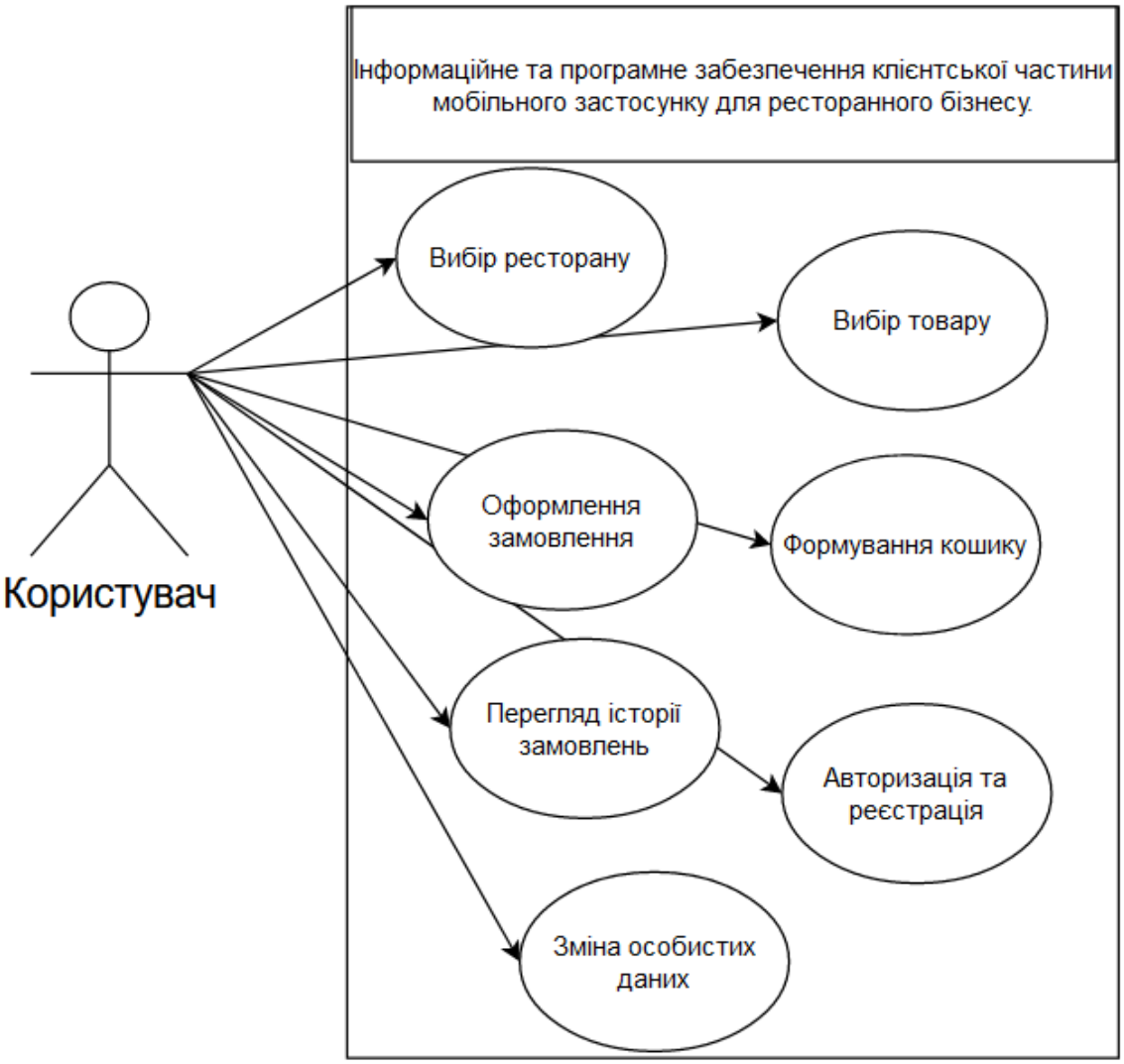


Рисунок 2.6 - Діаграма варіантів використання

# 3 РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОГО ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ КЛІЄНТСЬКОЇ ЧАСТИНИ МОБІЛЬНОГО ЗАСТОСУНКУ ДЛЯ РЕСТОРАННОГО БІЗНЕСУ

## 3.1 Розробка архітектури мобільного додатку

На рисунку 3.1 подано архітектуру 3 клієнтського рівня мобільного додатку. Так як мета реалізації заключається в створенні інформаційного та програмного забезпечення клієнтської частини мобільного застосунку для ресторанного бізнесу то саме 3 рівень розробки мобільного додатку підходить для аналізу. Для початку необхідно розділити клієнтський рівень на 2 підпроцеси: 1-й - це графічний інтерфейс або мобільний додаток, а 2-й - для бізнес-процесів [10].

Рівень графічного інтерфейсу забезпечує графічний інтерфейс користувача для взаємодії кінцевого користувача з сервером бази даних.

Для кінцевого користувача рівень графічного інтерфейсу є системою бази даних, і кінцевий користувач не має жодного уявлення про клієнтський рівень та сервер бази даних.

Переваги: масштабованість та покращена безпека.

Недоліки: висока вартість обслуговування.


Рисунок 3.1 - Архітектура 3 клієнтського рівня мобільного додатку

## 3.2 Програмна реалізація

### 3.2.1 Компоненти додатку

На початку розробки потрібно визначити головні компоненти при розробці клієнтської частини мобільного застосунку для ресторанного бізнесу. Серед безлічі розроблених елементів клієнтської частини необхідно зосередити увагу на наступних компонентах та модулях:

- компонент головної сторінки;
- компонент вибору оплати замовлення;
- контейнер виводу замовлення кошику замовлень;
- компонент підключення інформації в залежності від статусу користувача;
- модуль виводу інформації через мобільну мережу;
- компонент перевірки версії React Native.

Компонент головної сторінки відповідає за підключення всіх функціональних компонентів, які відображаються на головному екрані мобільного додатку. До цих компонентів можна віднести логотип додатку, навігаційну панель, банери з додатковою інформацією, секції з контентом, такі як категорії страв або спеціальні пропозиції. Також важливими для дизайну компонентами є кнопки дії для додавання елементів до кошика, та перехід до інших екранів. На рисунку 3.2 представлено частину коду на React Native, яка стосується створенню компонента головної сторінки. Повну реалізацію компоненту представлено в додатку В.

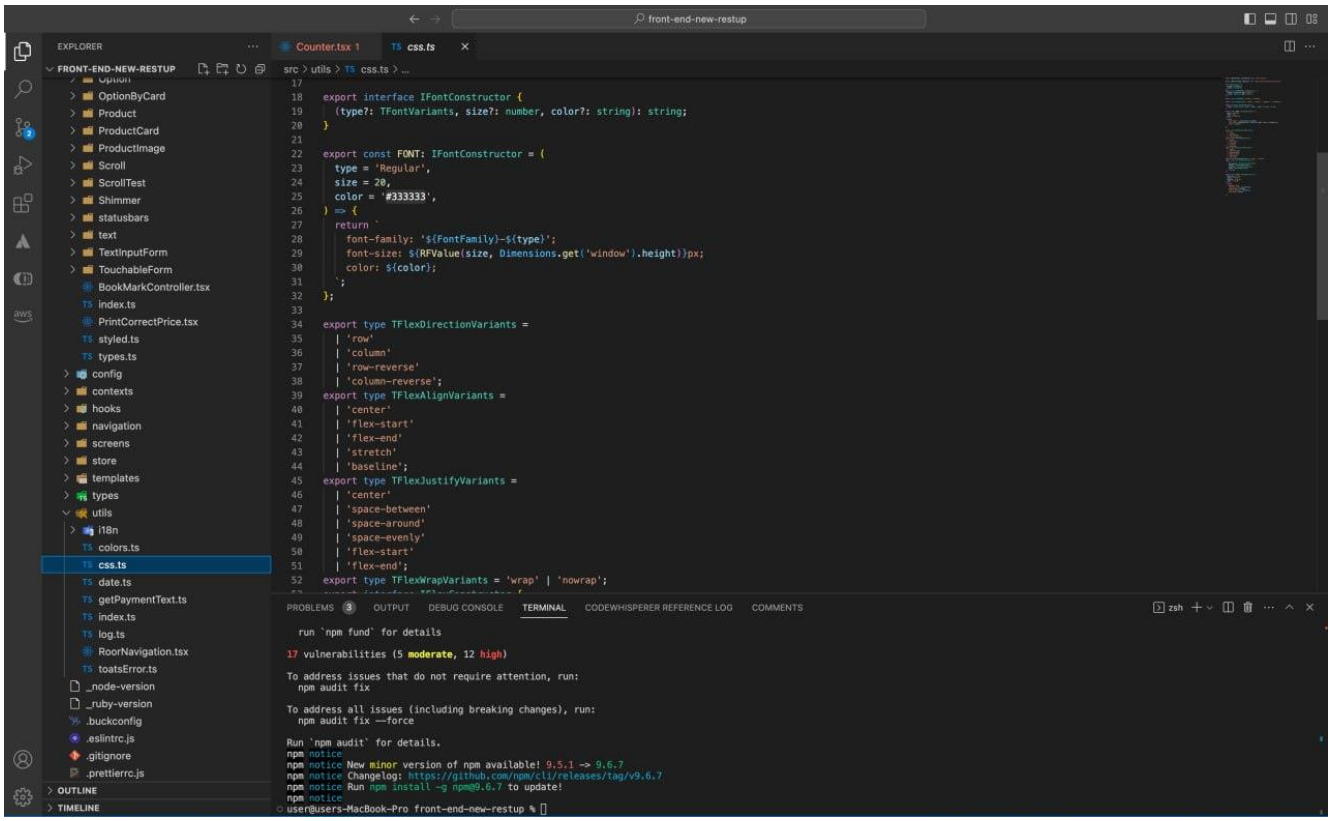


Рисунок 3.2 – Частина реалізації компоненту головної сторінки

Для постійного контролю версії додатку та фреймворку було створено компонент перевірки версій. Даний компонент допоможе за лічені хвилини перевірити, чи відповідає поточна версія фреймворку встановленим вимогам або обмеженням. Також дуже зручним моментом буде оновлення React Native до потрібної версії, якщо встановлена версія не відповідає вимогам системи. На рисунку 3.3 представлено частину коду реалізованого компоненту перевірки версій.

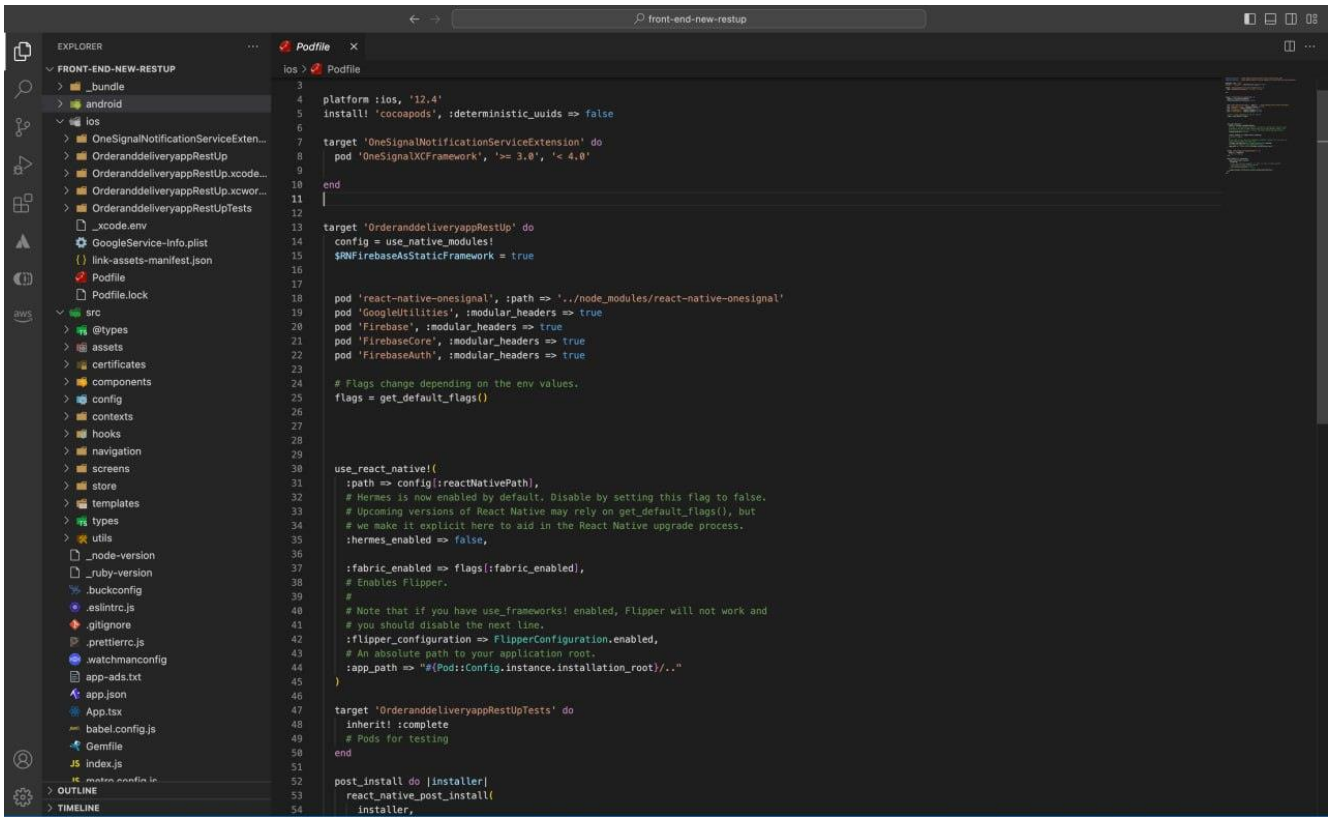


Рисунок 3.3 – Компонент перевірки версій

Також важливим елементом мобільного додатку є компонент вибору оплати замовлення, адже це відноситься до фінальної частини оплати замовлення. Саме цей компонент надає можливості вибору типу оплати. На рисунку 3.4 представлено частину коду реалізованого компоненту вибору оплати.

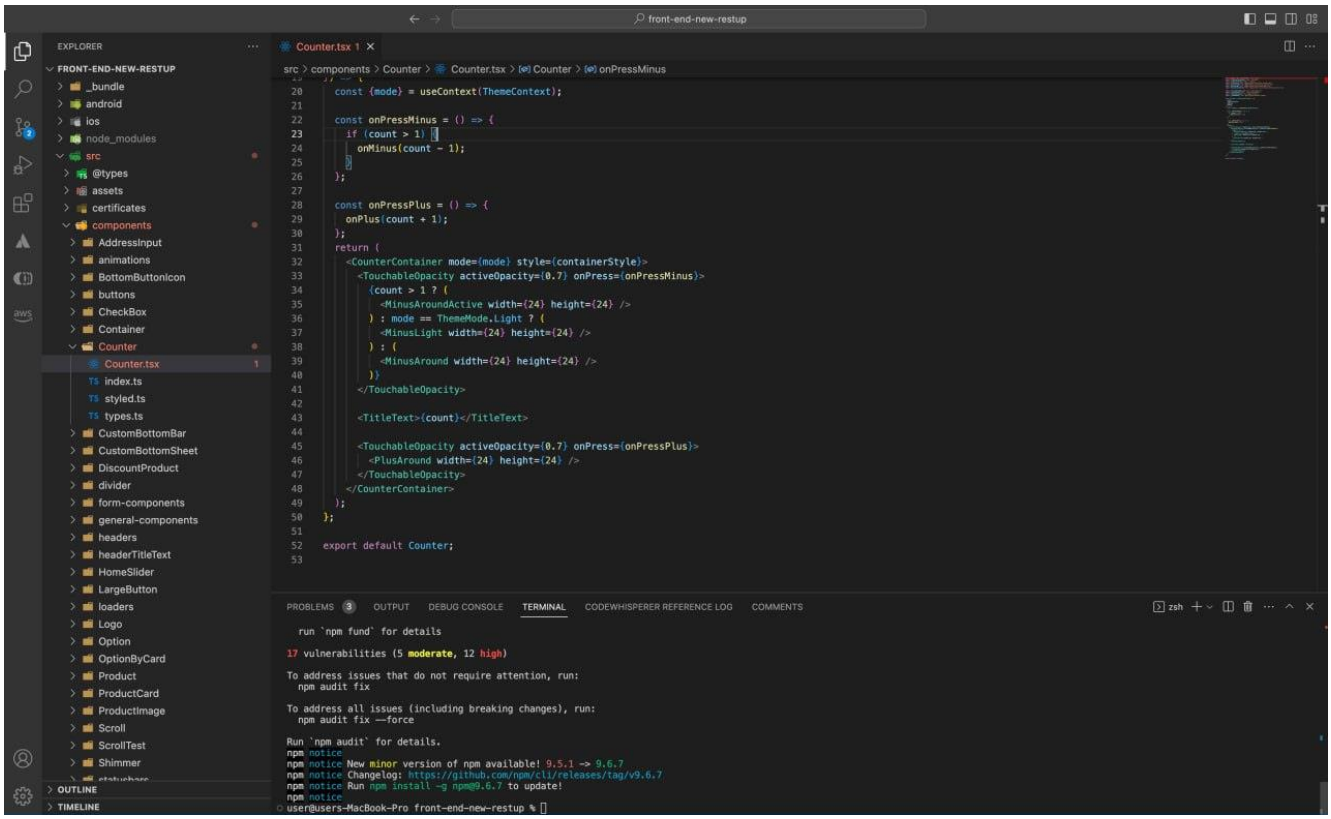


Рисунок 3.4 - Частина коду реалізованого компоненту вибору оплати

Одним з головних елементів додатку є модуль виводу інформації через мобільну мережу. Модуль містить в собі наступні компоненти та функціональність:

- компонент взаємодії з мобільною мережею ( загальний компонент, що дозволяє взаємодіяти з мобільною мережею, який може включати в себе стан підключення, вибір типу з’єднання та швидкості;
- компонент перевірки якості з’єднання та визначення його стану;
- компонент обробки помилок та їх оптимізацій (кешування даних, розподілення запитів для покращення продуктивності та реакція додатку на стан мережі);

На рисунку 3.5 представлено частину реалізації модулю виводу інформації через мережу.

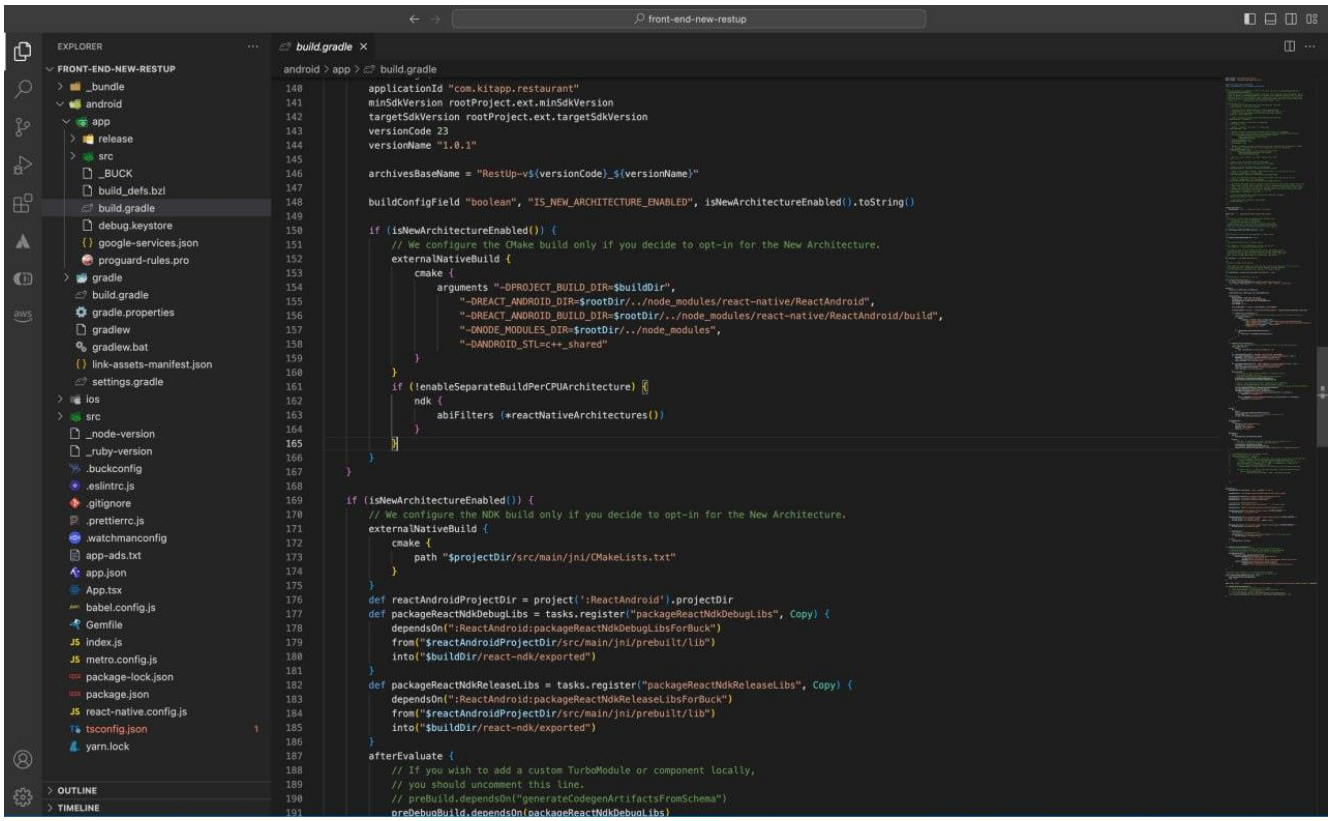


Рисунок 3.5 – Модуль виводу інформації через мережу

### 3.2.2 Дизайн додатку

Як тільки користувачеві подобається дизайн додатку, він переходить до оцінки його функціональних можливостей. Отже, щоб створити гарне перше враження, вам потрібно зосередитися на дизайні вашого мобільного додатку.

В якості інструменту для створення дизайну мобільного додатку було обрано додаток Figma. Головні переваги використання Figma проявляються в тому, що даний додаток працює в веб-браузері, що дозволяє отримати доступ до будь-якого проекту за лічені хвилини [11].

Також Figma надає можливість створення інтерактивних прототипів, які дозволяють перевірити функціональність додатку. На рисунку 3.6 представлено вигляд веб-версії додатку Figma з виконаним дизайном.

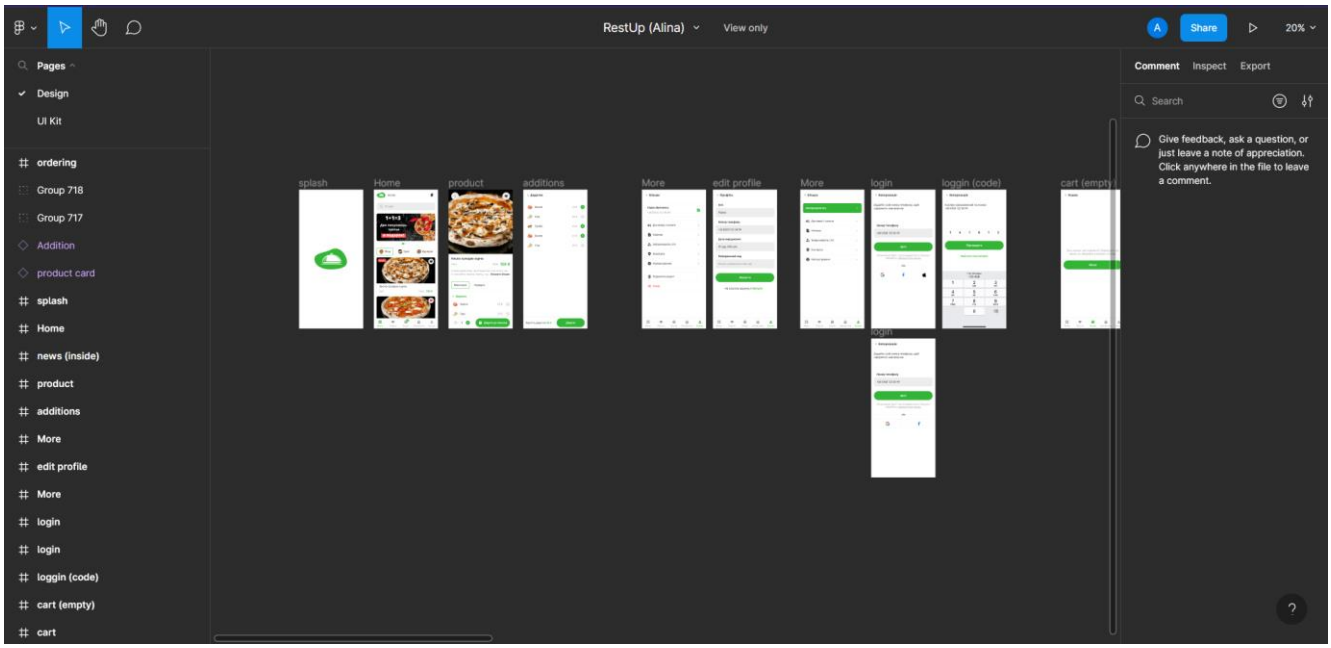


Рисунок 3.6 – Вигляд робочого середовища в програмі Figma

**Гарний дизайн додатку приваблює користувачів**

Перше, що вам потрібно, щоб привернути увагу користувачів, - це крутий візуальний ряд. Якщо ви можете створити мобільний додаток, що складається з чудових візуальних ефектів, інфографіки, правильних тематичних кольорів, він обов'язково приверне увагу користувачів з першого разу [12]. На рисунку 3.7 зображено реалізований дизайн екранів з інформацією про товар та головне меню.

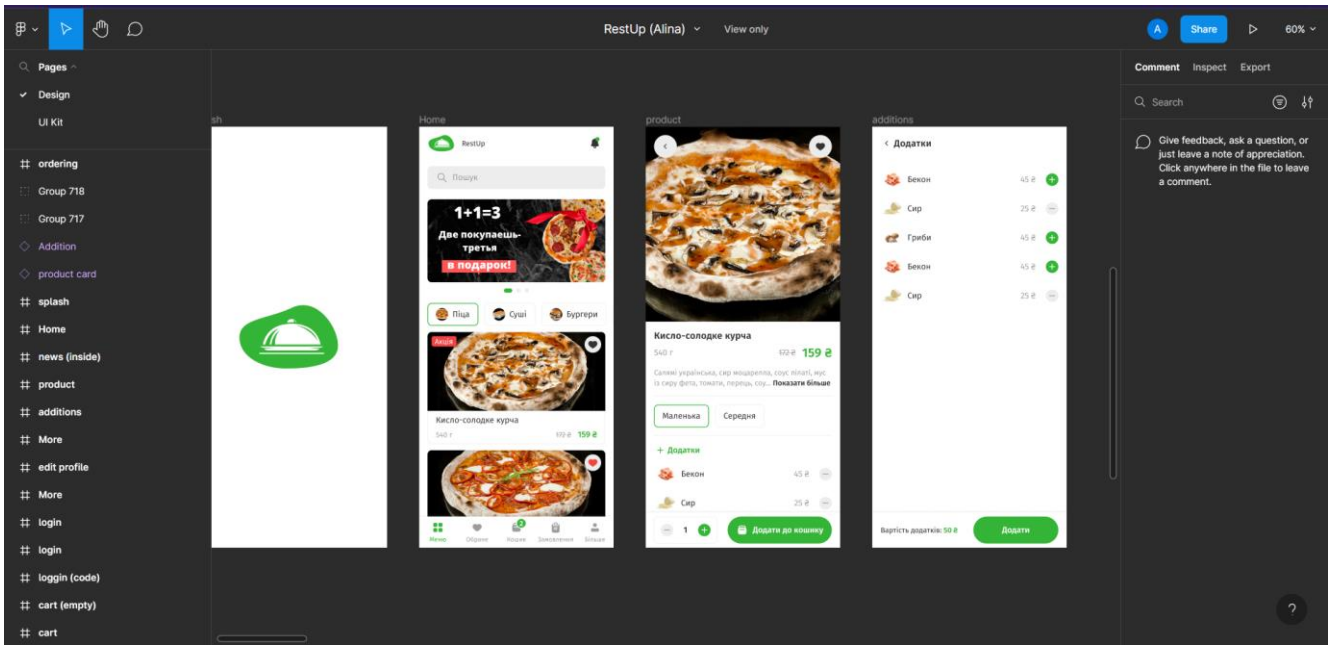

Рисунок 3.7 – Результат створення дизайну екранів «splash», «home», «product», «additions»

**Рішення для розробки мобільних додатків**

Ваша кінцева мета має полягати в тому, щоб користувачі залишалися у вашому мобільному додатку якомога довше. Щоб заохотити їх проводити більше часу у вашому додатку, вам потрібен UI/UX дизайн, який резонує з ними. Дизайн вашого додатку повинен бути настільки влучним, щоб користувачі не тільки завантажували його самі, але й рекомендували своїм друзям [13]. На рисунку 3.8 представлено результат створення екранів з додатковою інформацією та авторизацією.

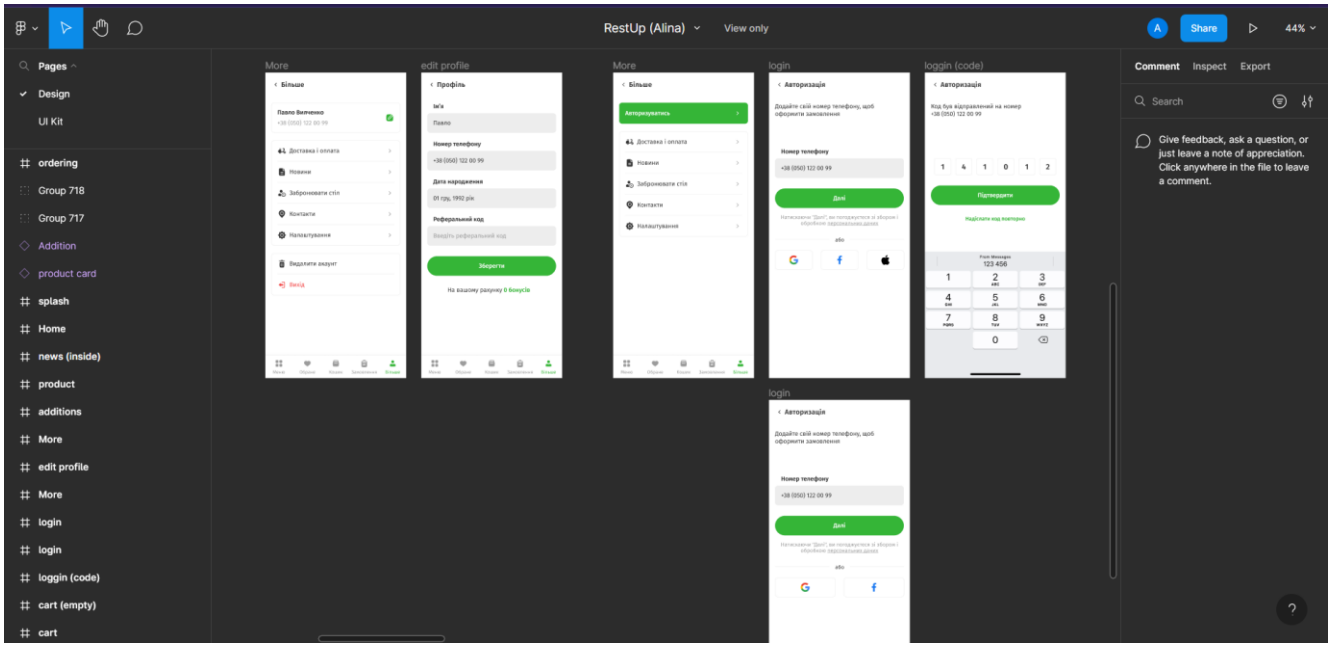


Рисунок 3.8 – Результат створення екранів з додатковою інформацією та авторизацією

**Адаптивний UI/UX допоможе зміцнити імідж бренду**

Мобільний додаток - це імідж бренду. Отже, це, безумовно, можливість для розробника зміцнити імідж власного бренду. Додаток із середнім користувацьким інтерфейсом не допоможе відставати від конкурентів. Потрібно створити щось більше, ніж середнє, щоб справити сильне перше враження на користувачів. Таким чином, мобільний додаток може посилити імідж вашого бренду [14].

За допомогою привабливого інтерфейсу без рекламної кампанії створити базу лояльних клієнтів. На рисунку 3.9 представлено вигляд екранів оформлення замовлення та додавання товару до кошика.

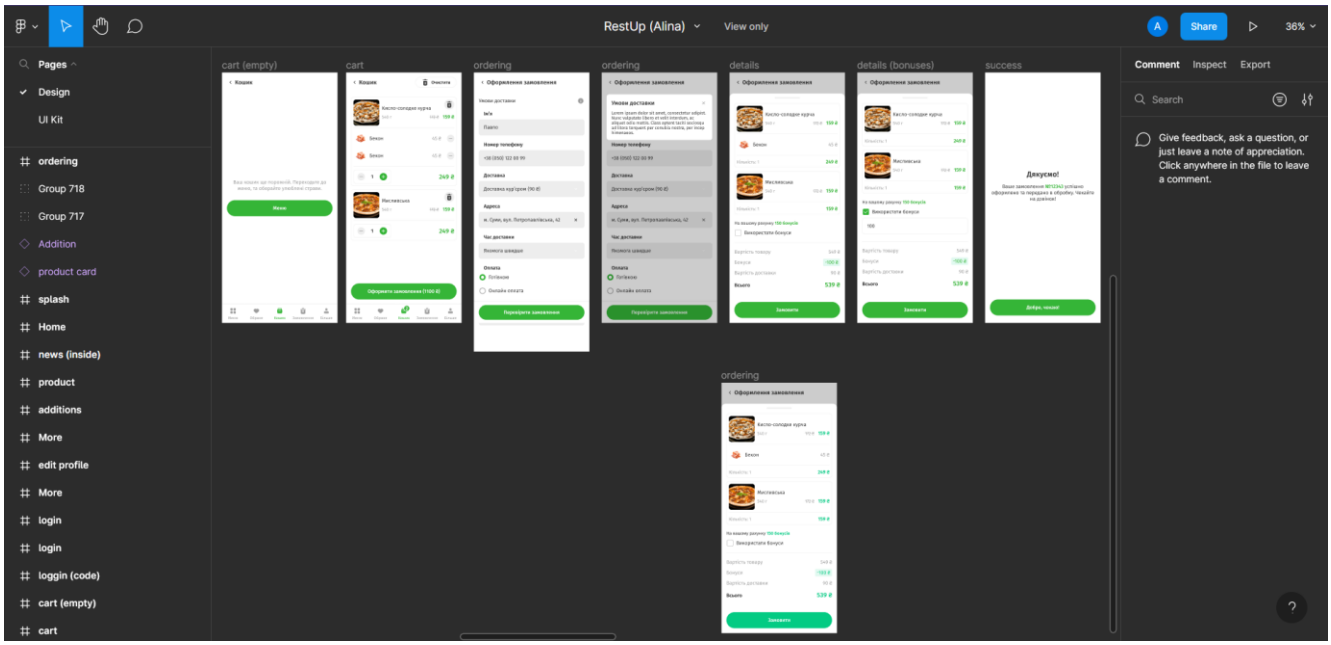


Рисунок 3.9 - Вигляд екранів оформлення замовлення та додавання товару до кошика

В мережі інтернет на 2023 рік існує багато безкоштовних шаблонів дизайну інтерфейсу для мобільних додатків але у більшості випадках ці шаблони вже давно інтегровані в більшість додатків на різну тематику. Тому для унікальності потрібно реалізовувати дизайн інтерфейсу з нуля. На рисунках 3.10 – 3.11 представлено результат створення дизайну екранів пошуку та перегляду обраних товарів, а також екранів оформлення замовлення.

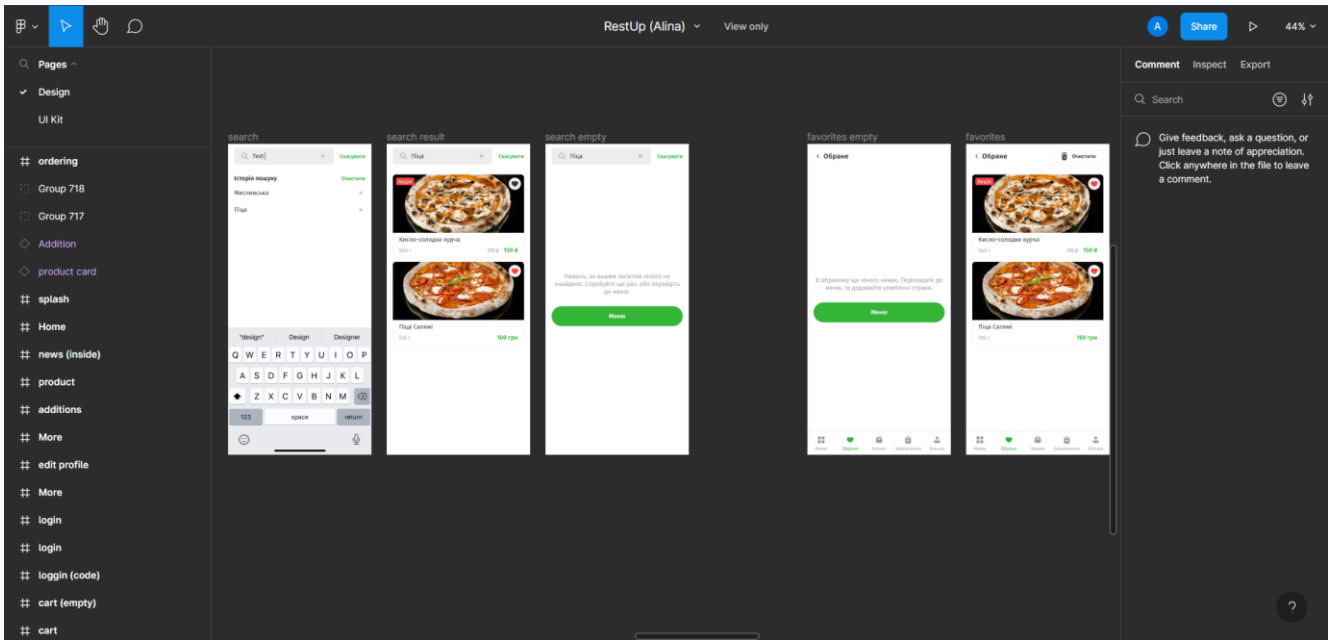


Рисунок 3.10 – Результат створення дизайну екранів та перегляду обраних товарів

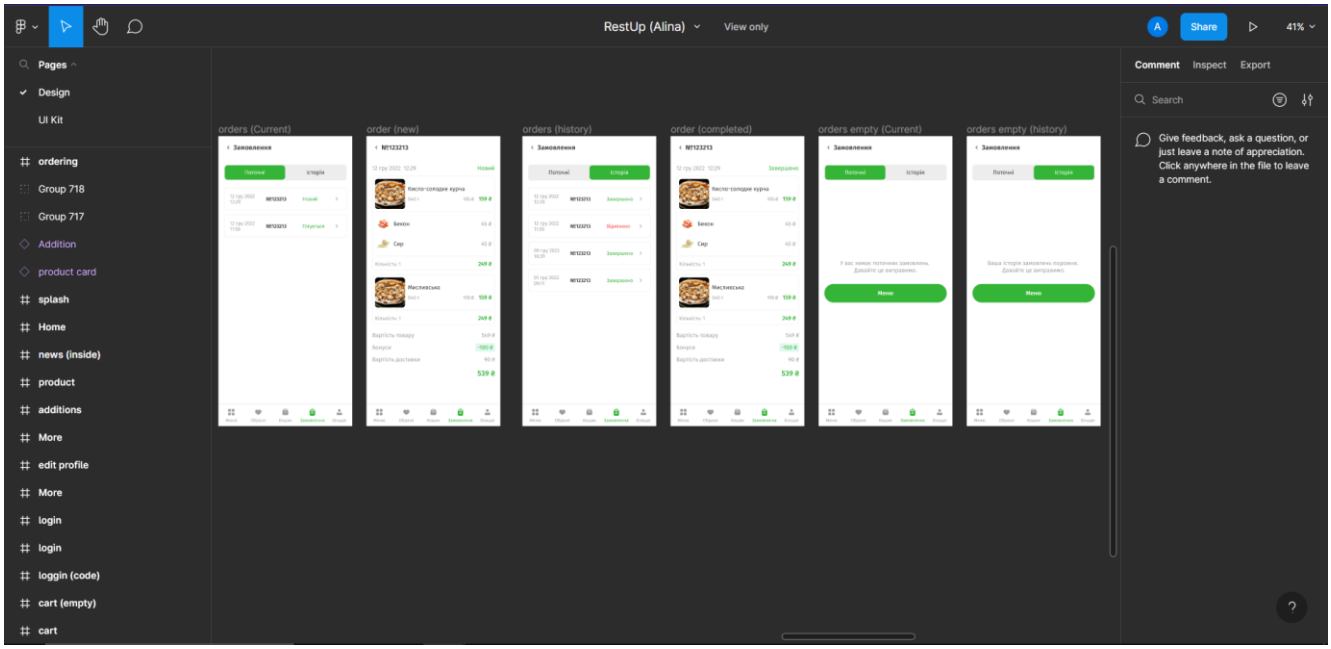


Рисунок 3.11 - Результат створення дизайну екранів оформлення замовлення

Фінальним етапом було створення дизайну екранів налаштувань, та корисної інформації для користувача ( інформація про доставку, контакти та інформація про обраний товар). На рисунку 3.12 представлено результат створення дизайну екранів з основною інформацією.

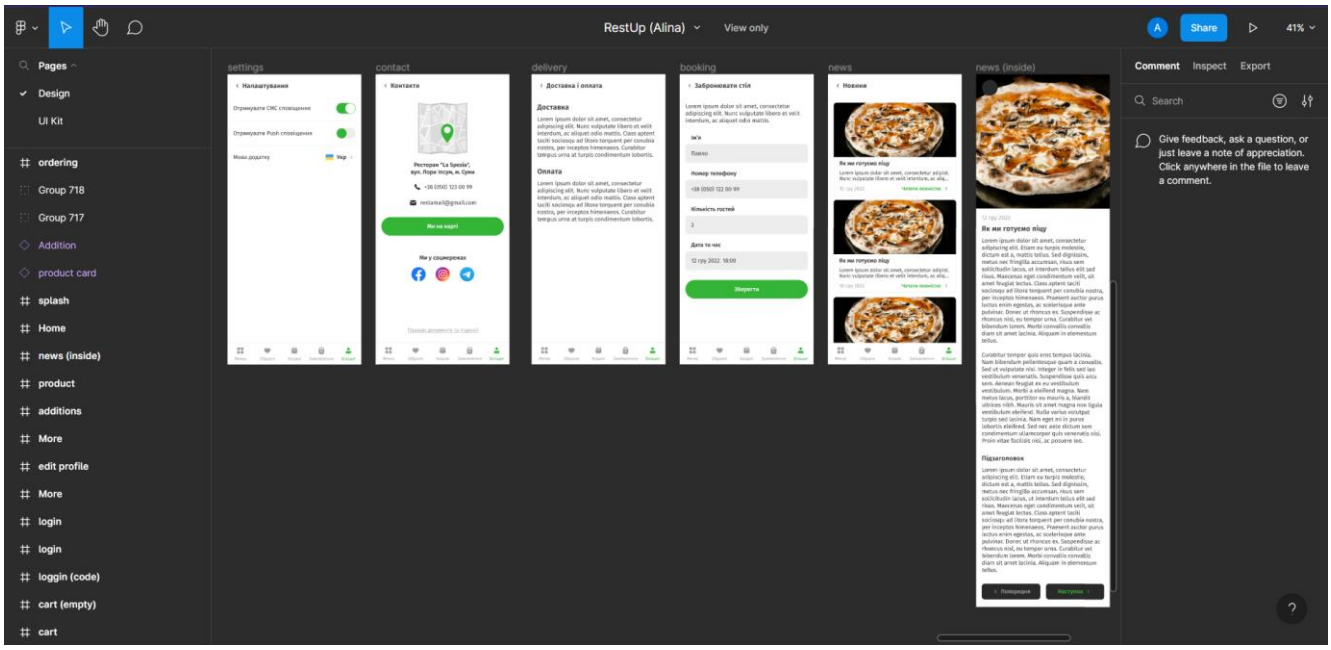


Рисунок 3.12 – Результат створення дизайну екранів з основною інформацією

Загалом, розроблений дизайн мобільного додатку надає чітке розуміння про те, як повинен виглядати та які функції повинен містити кінцевий результат. Розроблений шаблон містить власний логотип, який відображає концепцію ресторанного бізнесу та його основні можливості, а також відображає елементи, які показують структуроване меню. Використання привабливих кольорових схем з дотриманням стандартів забезпечують привабливий вигляд додатку та покращують користувацький досвід.

## 3.3 Розробка користувацької частини

UX - користувацький досвід для мобільних додатків так само важливий, як і для веб-сайту. За допомогою правильно реалізованого інтерфейсу, мобільний додаток, який може перетворити відвідувачів на постійних клієнтів, збільшуючи дохід у процесі. Тому варто зосередитися на важливості дизайну додатків і взяти до уваги важливі особливості дизайну користувацького інтерфейсу.

Важливість дизайну користувацького інтерфейсу та користувацького досвіду (UX) йдуть пліч-о-пліч, тобто вони не виключають одне одного. Навіть якщо UI/UX

не виключають один одного, це не означає, що це одне і те ж саме. Вони мають свої відмінності. Але UX більше зосереджений на тому, як все працює, тоді як мета дизайну користувацького інтерфейсу - зробити весь дизайн простішим, інтуїтивно зрозумілішим і привабливішим за допомогою візуальних елементів [15].

Якщо врахувати найкращі практики UI/UX з самого початку проекту, то не доведеться переробляти свій додаток знову і знову, що заощадить час і витрати. Крім того, якщо витратити значну кількість часу на дизайн інтерфейсу, то проблем під час і після запуску буде менше, оскільки клієнти не матимуть жодних проблем з продуктом. Адже бездоганний продукт не потребує постійних оновлень та підтримки.

Користувацька частина буде реалізована на основі створеного шаблону дизайну мобільного додатку.

Для початку необхідно реалізувати екран «splash», який буде містити в собі логотип бренду. Спочатку потрібно підготувати зображення логотипу в форматі, який підтримує React Native. В даному випадку це буде зображення формату .jpeg. Після цього необхідно виконати створення компоненту «SplashScreen», який буде відповідати за за відображення екрану запуску. Для створення компоненту використовувались такі властивості як «source», «setTimeout» та «setInterval». На рисунку 3.13 представлено результат створення екрану «splash». Всі результати продемонстровані на операційній системі IOS 16.0. Модель телефону – Iphone 11.

Рисунок 3.13 – Результат створення екрану «splash»

Наступним етапом було розроблено екран «Home». Для його розробки було створено новий файл «HomeScreen.js» основна задача якого відповідати за рендеринг екрану «Home». Також потрібно імпортувати всі компоненти, які необхідні для формування головної сторінки. Наприклад «View», «Text», «Image», «TouchableOpacity». Після цього необхідно описати структуру та підключити стилі, які будуть описані в файлі «StyleSheet.create». На рисунку 3.14 зображено вигляд створеного екрану«Home».

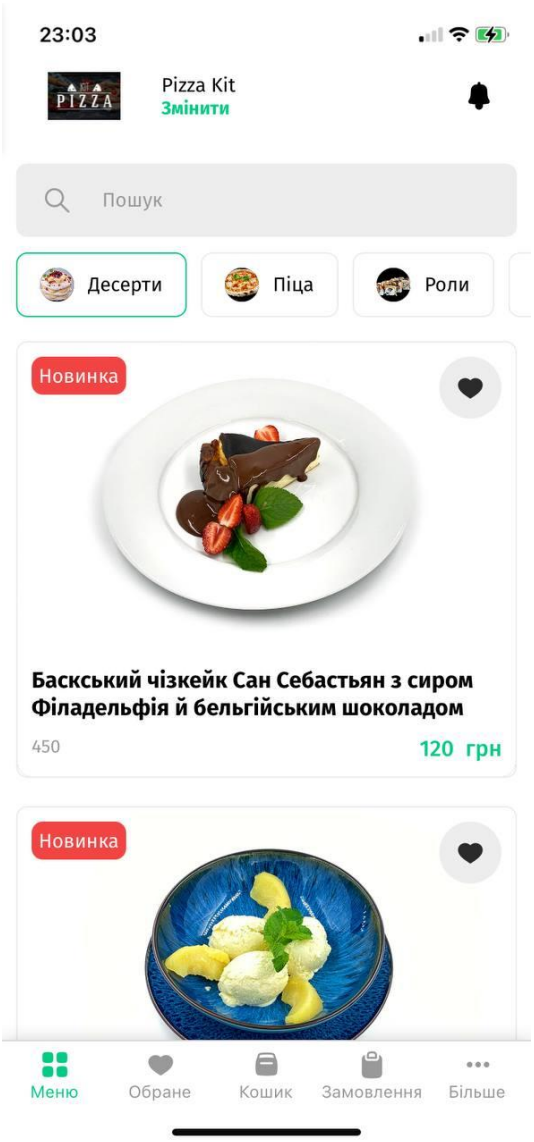


Рисунок 3.14 – Результат створення екрану «Home»

Наступний етап – створення екрану «Product», який буде містити основну інформацію про страву. Для цього використаємо алгоритм створення екрану «Home». Єдина відмінність створення полягає в підключених стилях, які повинні відповідати шаблону дизайну мобільного додатку. На рисунку 3.15 представлено результат створення екрану «Product».

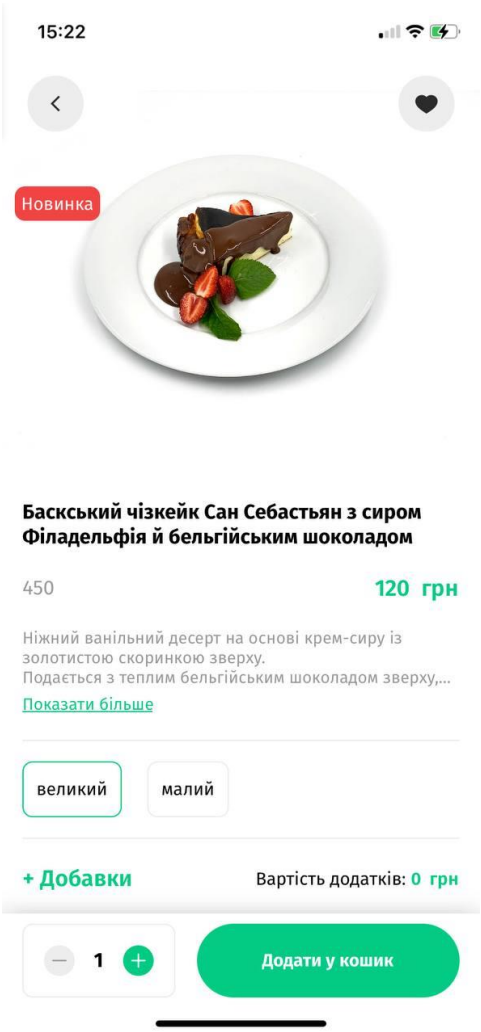


Рисунок 3.15 – Результат створення екрану «Product»

Також до екрану «Product» було створено додатковий екран з вибором додатків до товару. На рисунку 3.16 представлено результат створення екрану «Additions».


Рисунок 3.16 – Результат створення екрану «Additions»

Далі необхідно створити екрани авторизації користувача «Login». Для цього необхідно створити файл «LoginScreen.js», який буде відповідати за рендер екрану. Після цього необхідно імпортувати компоненти, які плануються для використання на екрані. В даному випадку це компоненти «View», «Text», «TextInput» та «Button». Також потрібно створити класовий компонент, який буде використовувати хуки.  Також як і в випадку з іншими екранами потрібно застосувати стилі за допомогою «StyleSheet.create». На рисунках 3.17 – 3.18 представлено результати створення екранів «Login».

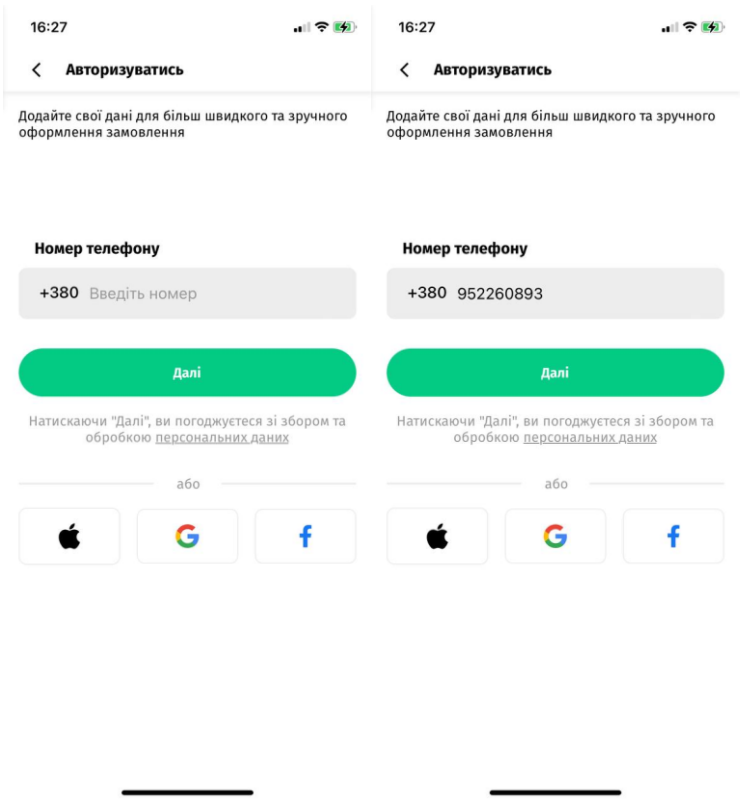


Рисунок 3.17 – Вигляд екрану «Login»

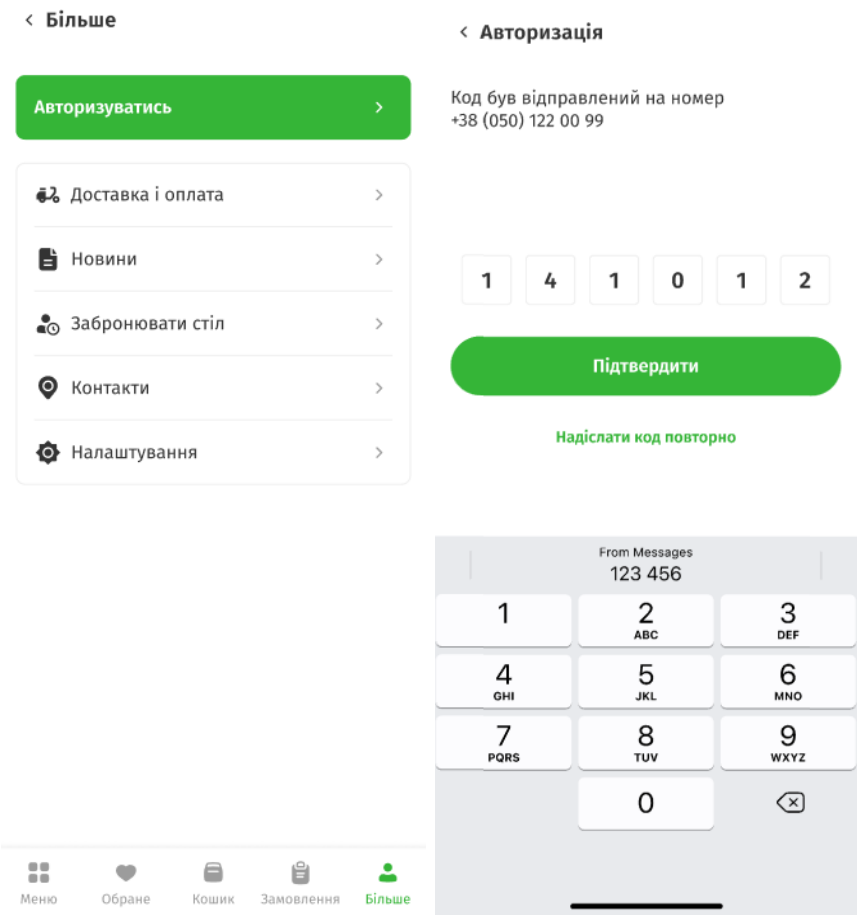

Рисунок 3.18 – Вигляд екрану підтвердження авторизації

Наступний етап – створення екранів кошику для додавання обраних страв для замовлення. За аналогом створення інших екранів було створено файл «CardScreen.js», який відповідає за рендеринг екранів. Також було додано компоненти для використання та хуки для використання функціональних компонентів. Для зберігання даних був використаний стан компонента, який потім передавався до батьківського компонента та сервісу для збереження даних. Також було розроблено систему бонусів для економії коштів постійних користувачів. На рисунках 3.19 – 3.20 представлено вигляд екранів «card (empty)» та «card».

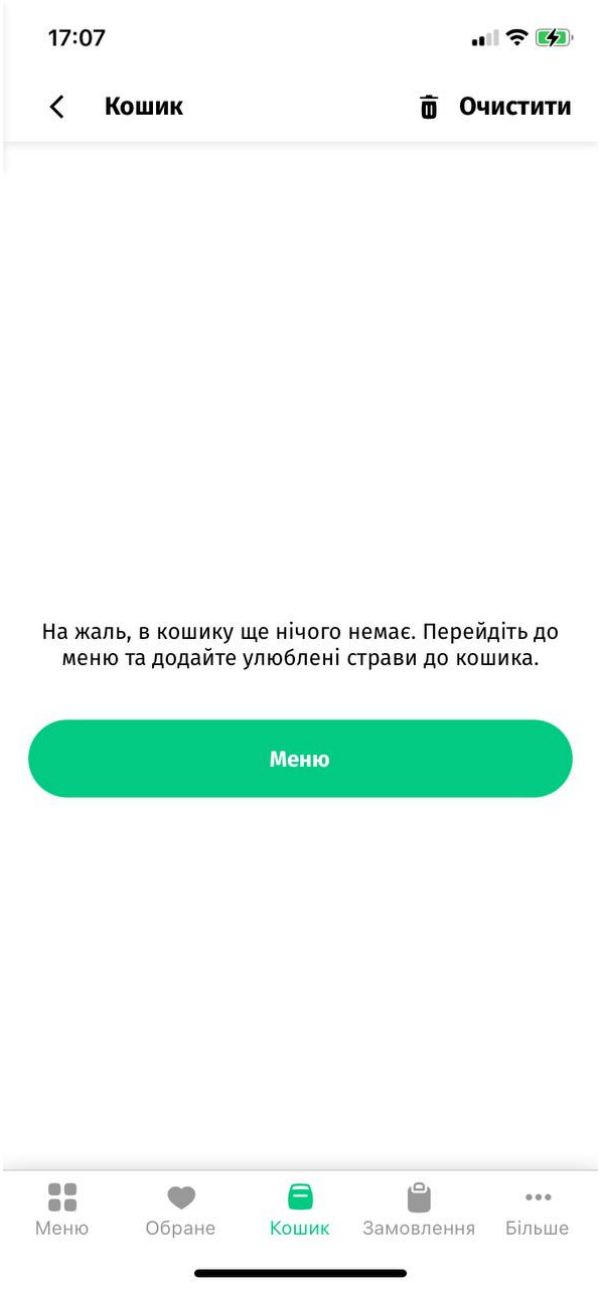

Рисунок 3.19 – Результат створення екрану «card (empty)»

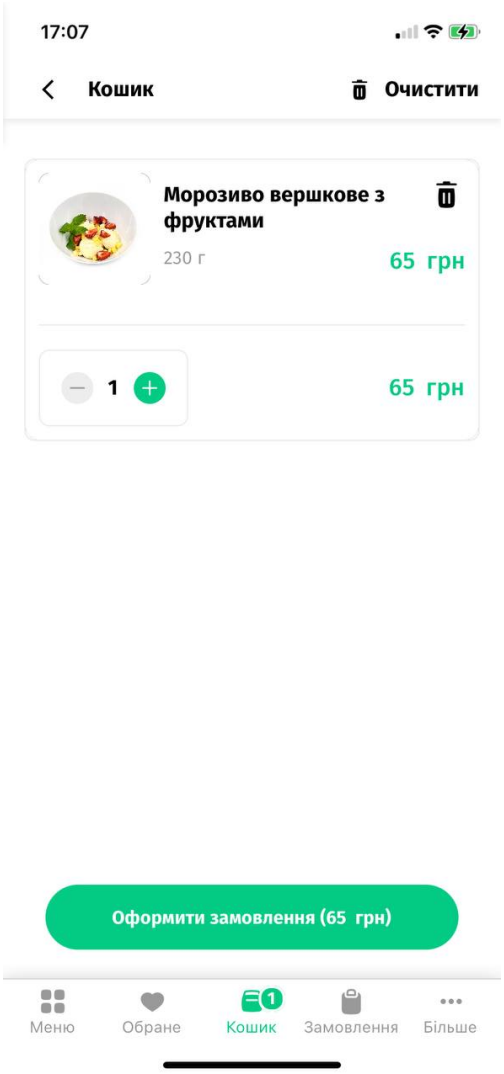


Рисунок 3.20 – Результат створення екрану «card»

Після того, як кошик сформований, необхідно створити екрани для оформлення замовлення, який буде містити в собі форму для заповнення. Якщо цю форму заповнює неавторизований користувач, то йому буде запропоновано виконати реєстрацію в системі. Для створення екранів використовувались хуки та компоненти, які будуть відповідати за обробку даних. На рисунках 3.21 – 3.22 представлено результати створення екранів для оформлення замовлення.

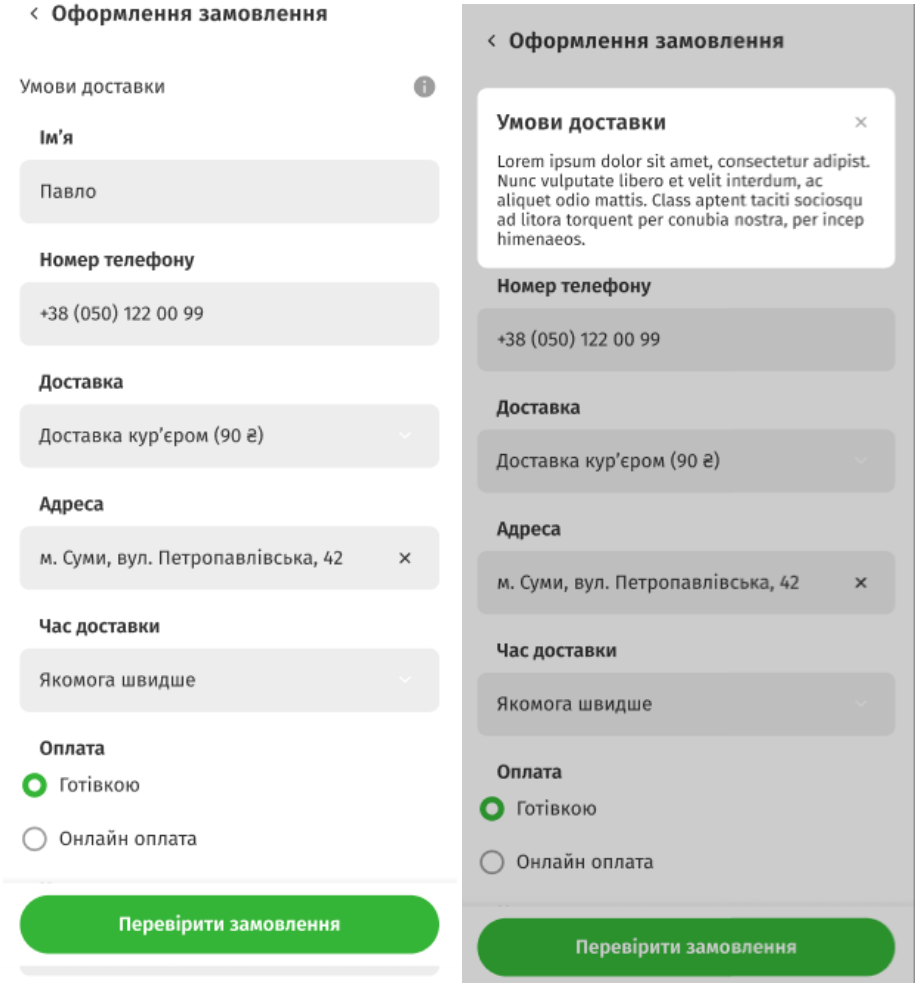


Рисунок 3.21 – Результат створення екранів «Ordering»

Також під час оформлення користувачу буде надано систему для використання бонусів. Результат створення екранів з системою представлено на рисунку 3.22.

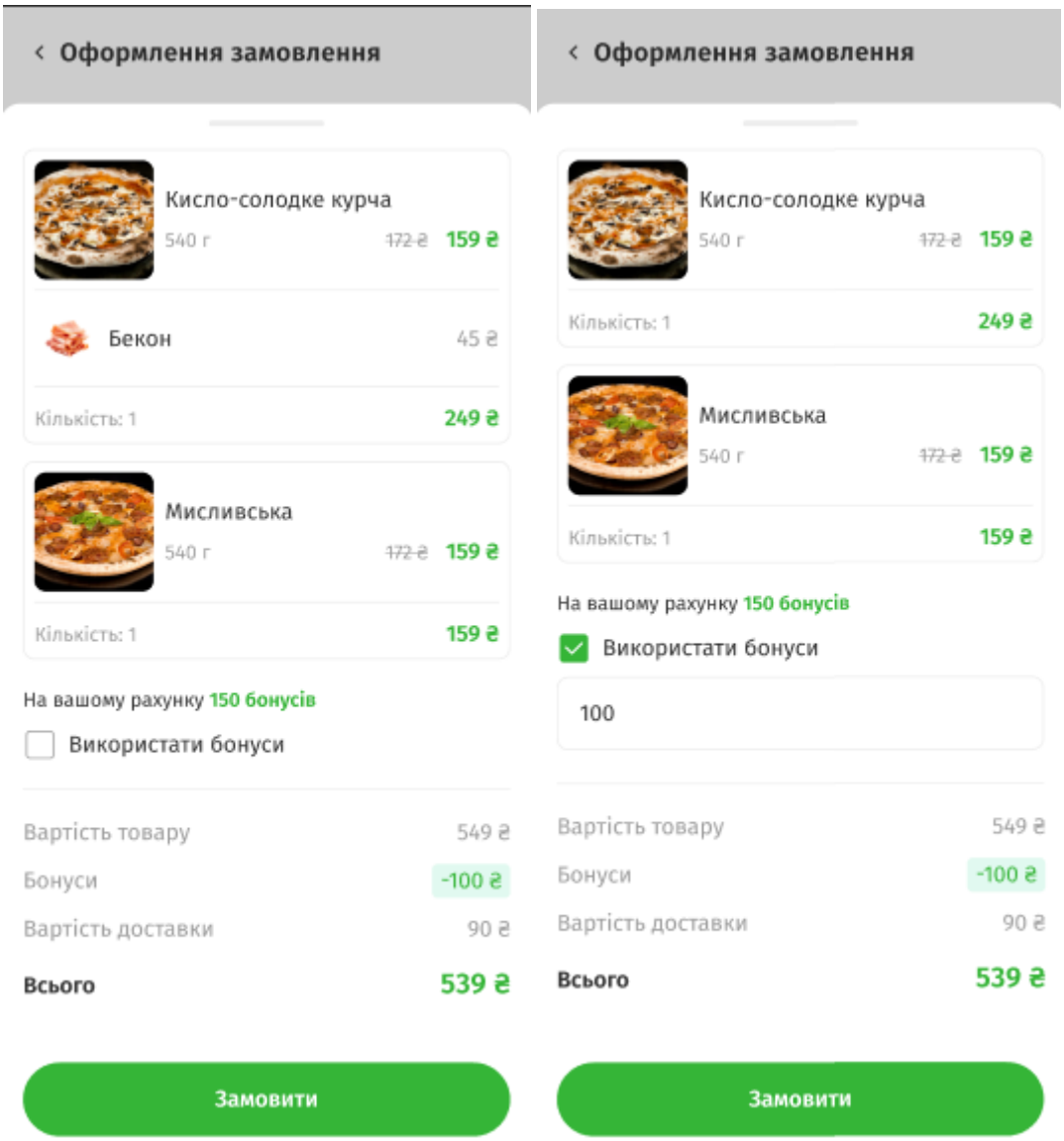


Рисунок 3.22 – Результат створення екранів з системою бонусів

Після виконаних операцій користувач отримає повідомлення про успішне замовлення (рис 3.24).


Рисунок 3.24 – Результат створення екрану «Succsess»

Система пошуку була створена для швидкого доступу до страв. Компонент «SearchScreen.js» відповідає за рендеринг екрану, компоненти «View», «Text», «Button» були обрані для використання на екрані. Також було створено класовий компонент «SearchScreen», який розширює «React.Component» який використовує хуки для функціональних компонентів. Метод «render» був застосований для опису структури та вигляду екрану пошуку. Результат створених екранів та результати роботи представлені на рисунку 3.25 – 3.26.

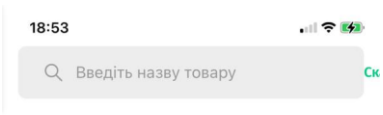


Рисунок 3.25 – Створений екран пошуку «Search»

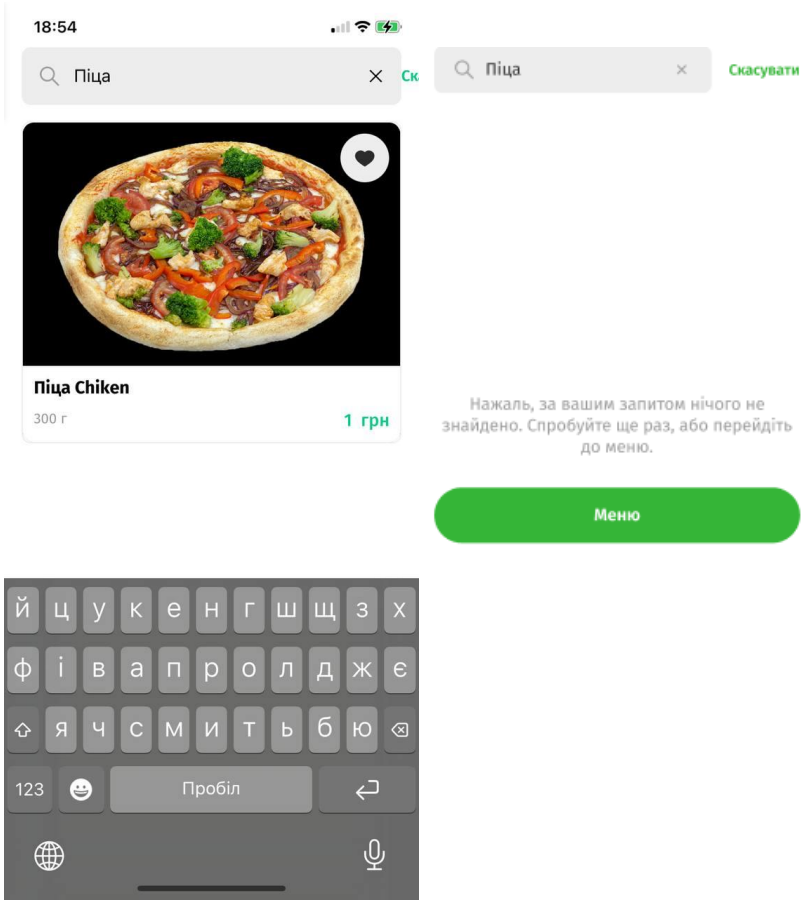


Рисунок 3.26 – Результати пошуку страви через екран «Search»

Для того, щоб користувач міг швидко замовити улюблені страви, було створено функцію для додавання обраних страв до обраних. Даний функціонал був реалізований за допомогою компонентів «SearchScreen», який розширює «React.Component» для використання хуків для функціональних компонентів. На рисунку 3.27 було представлено результат створення екранів «favorites» та «favorites empty».

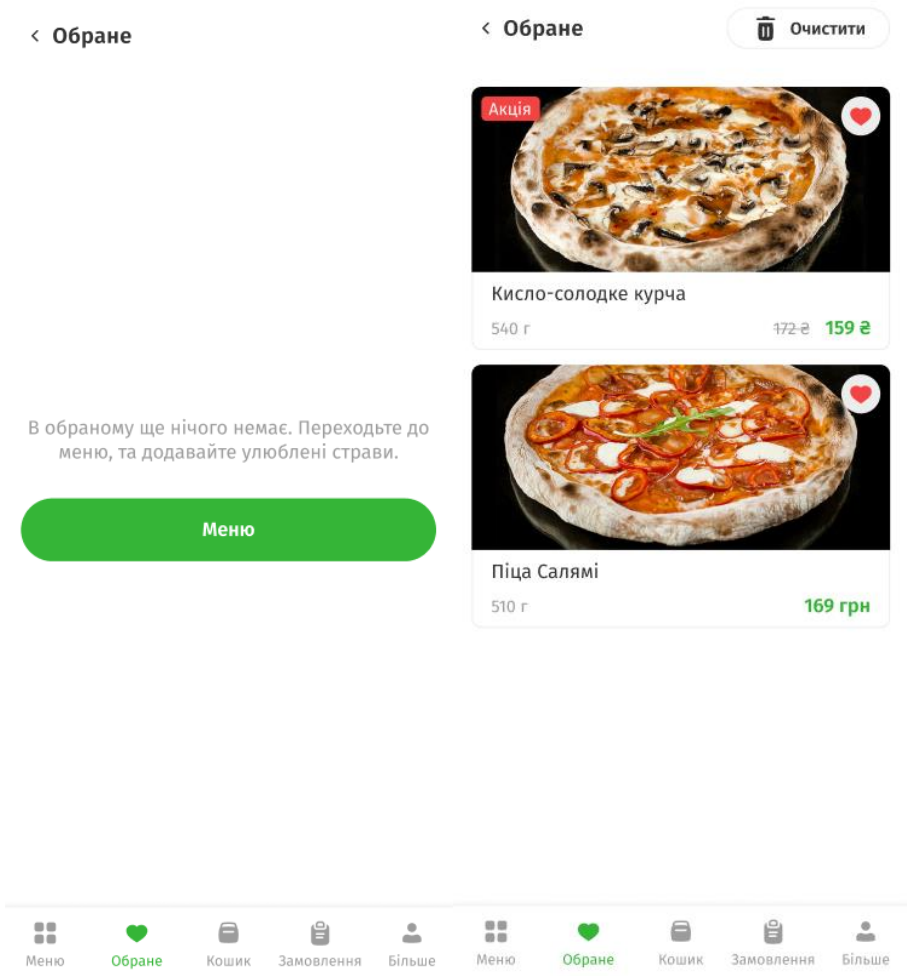


Рисунок 3.27 – Результат створення екранів «favorites» та «favorites empty»

Включення історії замовлень в мобільний додаток допоможе користувачам переглянути свої попередні замовлення без необхідності введення даних. Також подану інформацію можна використати для повторного замовлення. Також дана функція допоможе сформувати рекомендації для користувача що насамперед покращить персоналізацію обслуговування. На рисунках 3.28 – 3.29 представлено результат створення екранів для перегляду замовлення.

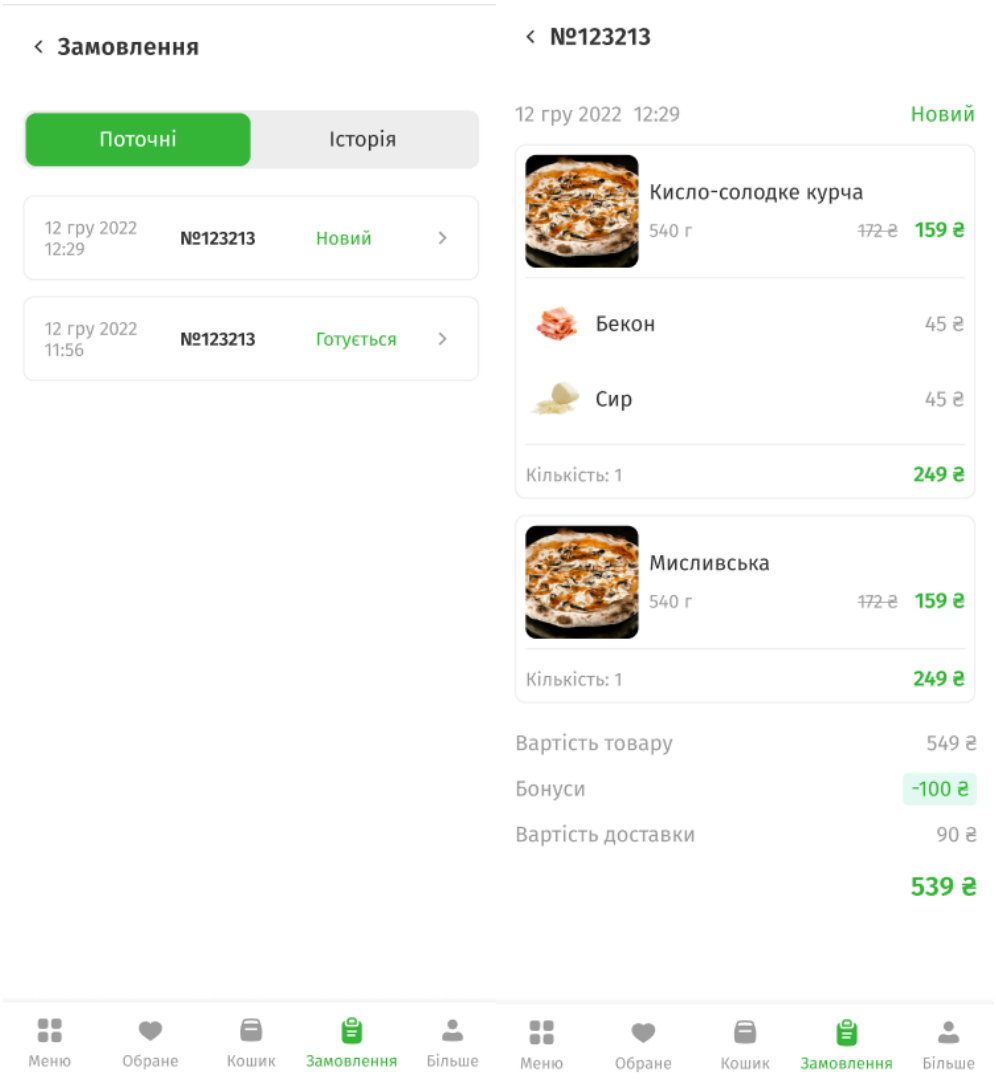


Рисунок 3.28 – Вигляд та вміст екрану «Поточні»

При переході до екрану замовлення, користувач зможе переглянути поточні та завершені замовлення. На натискання відповідного замовлення, користувач отримує детальну інформацію про завершений цикл оформлення та доставки їжі.

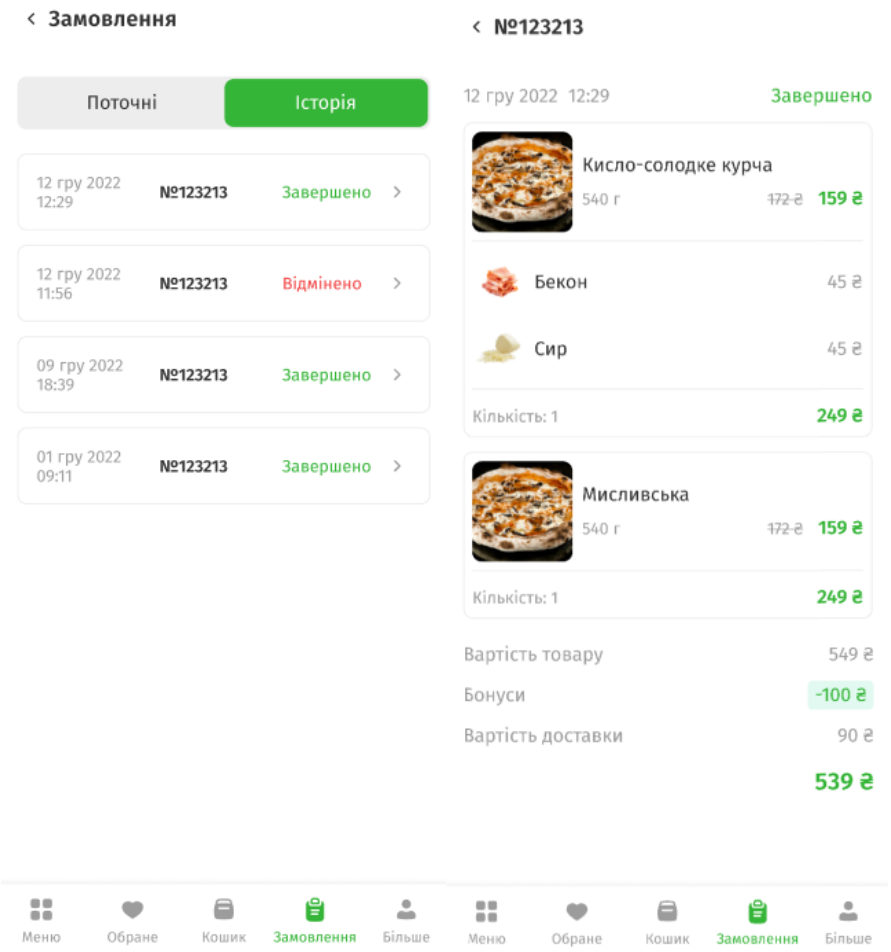


Рисунок 3.29 - Вигляд та вміст екрану «Історія»

В інформаційному та програмному забезпеченні клієнтської частини мобільного застосунку для ресторанного бізнесу система налаштувань відіграє важливу роль і забезпечує різні можливості та персоналізацію для користувачів. Налаштування надають змогу змінити додаток під власні вподобання і дозволяють змінювати контактні дані користувача. Також важливим елементом є функції відключення сповіщень та акцій, а також виконати вибір, які сповіщення надсилати для ознайомлення. На рисунку 3.30 представлено результат створення екрану «settings».

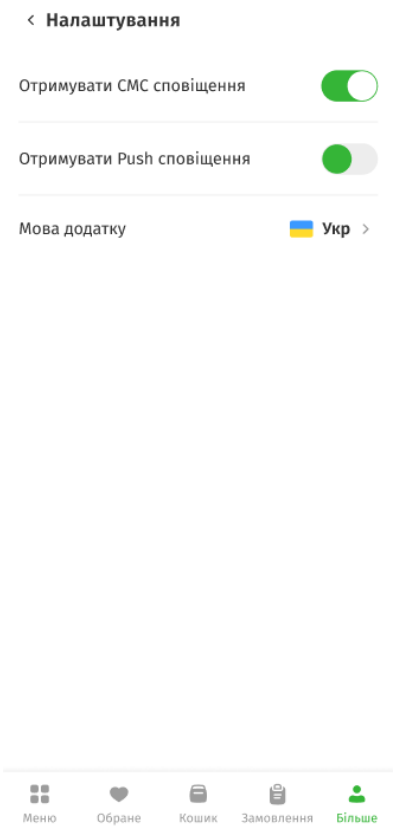


Рисунок 3.30 – Результати створення екрану «settings»

Розробка екрану з основною інформацією про діяльність ресторанного бізнесу дозволить користувачам легко отримати зв’язок з адміністрацією мобільного додатку. Також доступна операція «Забронювати стіл», яка полегшує процес замовлення та надає зручність користувачам. Контактні дані також допоможуть постачальникам продуктів та потенціальним партнерам отримати швидкий зв’язок. Це допоможе в розвитку бізнесу та створенню нових можливостей. На рисунках 3.31 – 3.32 представлено вигляд екранів «contact», «delivery» та «booking».

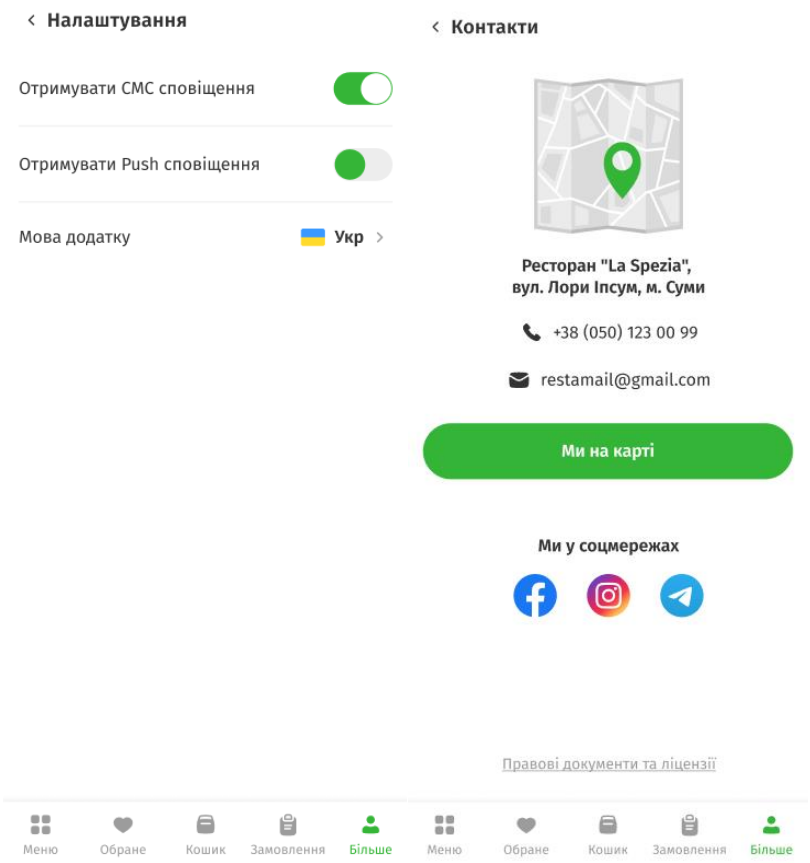


Рисунок 3.31 – Результат створення екранів «contact» та «delivery»

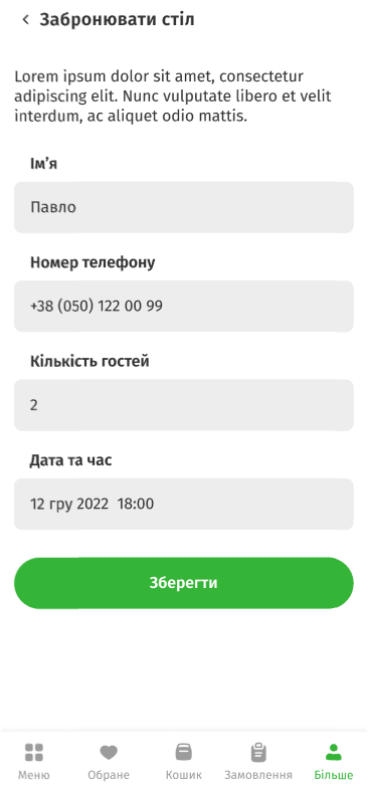


Рисунок 3.32 – Результат створення екрану «booking»

Фінальний етап – створення екранів для новин та повідомлень. Новини надаватимуть актуальну інформацію про знижки, промокоди та спеціальні пропозиції. Розділ новин також буде містити статті з області гастрономії, які зацікавлять клієнтів. Блок повідомлень буде сповіщати клієнта про зміну операцій замовлення. Це допоможе в актуалізації інформації. На рисунках 3.33 – 3.34 представлено результат створення екранів «news» та «nofitication».


Рисунок 3.33 – Результат створення екрану «news»

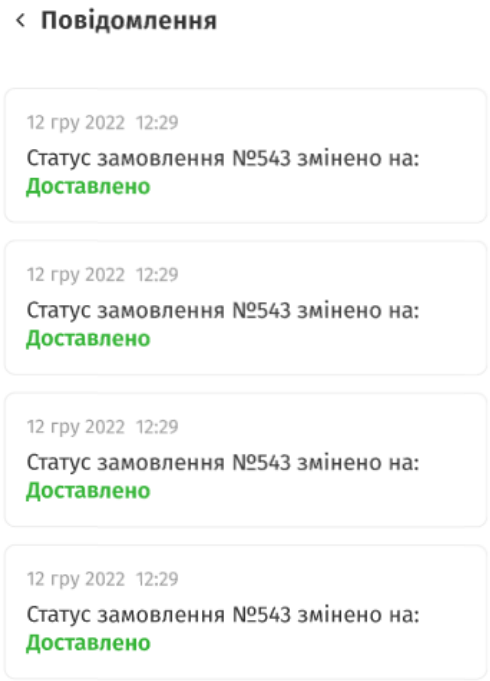


Рисунок 3.34 – Результат створення екрану «nofitication»

# ВИСНОВОК

В результаті виконання кваліфікаційної роботи бакалавра , було розроблено клієнтську частину мобільного додатку для підтримки ресторанного бізнесу «RestUp». Мобільний додаток має простий і зрозумілий у використанні інтерфейс, що дозволяє швидко і легко адаптуватися для ефективної роботи з ним. Це означає, що навіть недосвідчений користувач зможе з легкістю застосовувати даний ресурс, щоб знайти і замовити необхідну йому страву. Також була розроблена адміністративна панель, за допомогою якої в будь-який час можна видалити або додати товар та змінити графічне оформлення сайту.

Даний сервіс може використовуватися з будь-якої точки країни, що безсумнівно дуже зручно, оскільки користувачам не буде необхідним їхати до ресторану, вони зможуть оглянути і замовити товар прямо з дому. Клієнтська частина мобільного додатку реалізована в повному обсязі та у відповідності з усіма заданими вимогами і нормами. Усі поставлені завдання виконані автором.

# СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1 7 Reasons why business need mobile apps [Електронний ресурс]– Режим доступу до ресурсу:https://www.techtarget.com/searchcustomerexperience/tip/7-reasons-why-businesses-need-mobile-apps.

2 13 benefits of mobile apps for business You need to know [Електронний ресурс]– Режим доступу до ресурсу: https://decode.agency/article/benefits-mobile-apps-business/.

3 Biggest Challenges in Mobile App Development in 2023 [Електронний ресурс] – Режим доступу до ресурсу: https://www.linkedin.com/pulse/biggest-challenges-mobile-app-development-2023-mindinventory-1f/.

4 TouchBistro: Restaurant Management POS System [Електронний ресурс] – Режим доступу до ресурсу: https://www.touchbistro.com/.

5 Zuppler: Online food ordering solution for restaurants [Електронний ресурс] – Режим доступу до ресурсу: https://www.zuppler.com.

6 Toast | Restaurant Point of Sale & Management System [Електронний ресурс] – Режим доступу до ресурсу: https://pos.toasttab.com/.

7 Develop a Mobile App With React Native [Електронний ресурс] – Режим доступу до ресурсу: https://javascript.plainenglish.io/develop-a-mobile-app-with-react-native-85d7228e6d2b.

8 IDEF0 Diagrams [Електронний ресурс] – Режим доступу до ресурсу: https://www.conceptdraw.com/solution-park/software-idef0.

9 Use Case Diagram [Електронний ресурс] – Режим доступу до ресурсу: https://www.smartdraw.com/use-case-diagram/.

10 2-Tier and 3-Tier Database Architecture [Електронний ресурс] – Режим доступу до ресурсу: https://medium.com/analytics-vidhya/2-tier-and-3-tier-database-architecture-6005c6527ee4.

11 Офіційний web-додаток Figma [Електронний ресурс] – Режим доступу до ресурсу: https://www.figma.com/files/recent?fuid=1014917051281648324.

12 Why Design is the Most Important Factor in a Mobile App Development? [Електронний ресурс] – Режим доступу до ресурсу: https://www.ishir.com/blog/9633/why-design-is-the-most-important-factor-in-a-mobile-app-development.htm.

13 Importance of Designing in Mobile App Development [Електронний ресурс] – Режим доступу до ресурсу: https://www.geeksforgeeks.org/importance-of-designing-in-mobile-app-development/.

14 Importance of UI/UX Design in the Development of Mobile Apps [Електронний ресурс] – Режим доступу до ресурсу: https://www.thepsi.com/importance-of-ui-ux-in-the-development-of-mobile-apps/.

15 The Importance of User Interface Design in Mobile Applications [Електронний ресурс] – Режим доступу до ресурсу: https://invozone.com/blog/the-importance-of-user-interface-design-in-mobile-applications/.

16 Постановка цілей по SMART [Електронний ресурс] – Режим доступу до ресурсу: https://pdatu.edu.ua/images/vihovna-robota/psiholog/ps10.pdf.

17 Що таке WBS? Повне керівництво по структурі роботи [Електронний ресурс] – Режим доступу до ресурсу: https://uk.education-wiki.com/8908957-what-is-wbs.

18 Організаційна структура управління: типи і характерні особливості [Електронний ресурс] – Режим доступу до ресурсу: https://pidru4niki.com/15880315/menedzhment/organizatsiyna_struktura_upravlinnya_tipi_harakterni_osoblivosti.

**ДОДАТОК А**

# Технічне завдання
# на створення програмного продукту

«Інформаційне та програмне забезпечення клієнтської частини мобільного застосунку для ресторанного бізнесу»

**2023**

# 1 ПРИЗНАЧЕННЯ Й МЕТА СТВОРЕННЯ ДОДАТКУ

## 1.1 Призначення мобільного додатку

Мобільний додаток повинен представляти доступні послуги обраного користувачем ресторану, знайомити з асортиментом а також надавати основну інформацію про страви.

## 1.2 Мета створення веб-додатку

Мета мобільного додатку – надання можливості користувачу на основі отриманої інформації виконувати замовлення страв та переглядати весь наданий асортимент ресторану.

## 1.3 Цільова аудиторія

Цільова аудиторія мобільного додатку включає наступні групи користувачів:

– власники та управлінці ресторанів (моніторинг продажів, управління меню та промоціями, а також отримання аналітики щодо роботи ресторану);
– офіціанти та персонал обслуговування (прийом замовлень, обробка платежів, створення комунікації з кухнею та забезпечення швидкого та ефективного обслуговування клієнтів);
– клієнти ресторанів (замовлення їжі, перегляд меню, здійснення оплати, резервування столиків, отримання знижок та спеціальних пропозицій);
– менеджери ресторанів (централізоване керування та моніторинг ресторану, управління запасами, перегляд аналітики та звітність)

# 2 ВИМОГИ ДО МОБІЛЬНОГО ДОДАТКУ

## 2.1 Вимоги до структури й функціонування мобільного додатку

Для ефективної підтримки ресторанного бізнесу мобільний додаток розробляється з метою надання зручного та доступного інструменту для ресторанів і їх клієнтів. Цей додаток буде доступний для завантаження з платформ Play Market

та App Store, що дозволить користувачам з різних мобільних пристроїв отримати доступ до його функціоналу.

Мобільний додаток буде складатися з взаємопов'язаних розділів, які будуть чітко виконувати свої функції. Наприклад в додатку передбачена можливість перегляду меню, замовлення їжі, оплати, резервування столиків тощо.

Створення мобільного додатку забезпечить зручну та швидку взаємодію між рестораном і його клієнтами, покращить обслуговування та забезпечить ефективне управління бізнесом.

## 2.2 Вимоги до персоналу

У контексті мобільних додатків, персонал повинен мати розуміння процесу встановлення та оновлення додатків на мобільних пристроях, знання про основні функції та налаштування мобільних операційних систем, а також уміння використовувати додатки та розуміти їхні інтерфейси.

Загальні навички роботи з персональним комп'ютером і мобільним телефоном є важливими для забезпечення ефективного використання мобільного додатку та його інтерфейсу з боку персоналу. Такі навички дозволять забезпечити гладку роботу додатку, оперативно реагувати на проблеми та забезпечити задоволення потреб користувачів.

## 2.3 Вимоги до стилістичного оформлення мобільного додатку

Інтерфейс клієнтської частини мобільного застосунку для ресторанного бізнесу повинен забезпечувати зрозуміле та логічне представлення розміщеної на ньому інформації, а також легкий перехід між розділами. Основна мета інтерфейсу полягає в тому, щоб користувачі могли швидко та зручно знаходити необхідну інформацію.

# 3 ОСНОВНІ ВИМОГИ

## 3.1 Структура інформаційної системи

Мобільний додаток повинен складатися з наступних розділів:

- екран «вибрати ресторан» – містить інформацію про доступні ресторани;
- екран «меню» – містить доступні страви обраного ресторану;
- екран «обране» – містить страви які найбільше подобаються користувачу;
- екран «кошик» – містить страви, які додав користувач для замовлення;
- екран «замовлення» – містить форму для замовлення страви;
- екран «більше» – містить основну інформацію для ознайомлення та можливість користувачу виконати авторизацію;

## 3.2 Навігація

При відкритті мобільного додатку, користувачеві повинна бути доступна логічно організована структура з розділами, які відображають основні функції та можливості мобільного додатку. Це розділи, які присвячені замовленням, меню, резервуванню столиків, акціям та іншим важливим аспектам роботи ресторану.

Кожен елемент або опція, доступна для вибору в інтерфейсі, повинна мати зрозумілу та інтуїтивно зрозумілу іконку або назву, щоб користувач міг швидко зрозуміти, що відбудеться після вибору цього елемента. При виборі будь-якого з елементів, мобільний додаток повинен негайно завантажувати відповідну інформацію, щоб користувач міг швидко ознайомитись з нею та здійснити потрібні дії.

## 3.3 Вимоги до програмного забезпечення

Програмне забезпечення клієнтської частини повинне задовольняти наступним вимогам:

Версія системи мобільного пристрою: Android – 7.0 та вище, IOS – 9 та вище;

Апаратне забезпечення для розробки мобільного додатку повинне відповідати таким вимогам:

– JavaScript 1.8.5;

– Фреймворк ReactNative 18.2.0;

– Також додаток повинен відповідати всім вимогам для проходження модерації для розміщення на PlayMarket та AppStore.

## 3.4 Функціональні вимоги

– Наявність системи пошуку та вибір категорії з можливістю застосування фільтрів.

– Оформлення замовлення та вибір системи доставки та оплати

– Додавання обраних страв до кошику та обраних.

– Створення особистого кабінету користувача з можливістю редагування інформації та видалення особистих даних.

– Методи зв’язку з адміністрацією сайту.

– Можливість налаштовувати мобільний додаток під власні потреби.

## 3.5 Наповнення сайту

Початкова розробка та наповнення контентом мобільного додатку здійснюється спільними зусиллями Виконавця і Замовника. Замовник надає всі необхідні матеріали для створення вмісту додатку. Для реалізації мобільного додатку використовується фреймворк React Native.

Виконавець відповідає за технічну реалізацію додатку та розробку його функціональності, використовуючи фреймворк React Native. Замовник в свою чергу забезпечує необхідний контент для наповнення додатку. Це можуть бути тексти, зображення, відео, меню ресторану, акційні пропозиції та інші матеріали, необхідні для представлення інформації користувачам.

Такий підхід до розробки та наповнення контентом дозволяє забезпечити високу якість і відповідність додатку потребам Замовника, а також ефективну співпрацю між сторонами. Виконавець забезпечує технічну експертизу, а Замовник вносить свої знання та матеріали, щоб забезпечити належний контент додатку.

# ДОДАТОК Б

## Деталізація мети проекту методом SMART

Метод SMART (Specific, Measurable, Achievable, Relevant, Time-bound) є відомим інструментом управління проектами, який використовується для постановки мети та формулювання конкретних, вимірюваних, досяжних, релевантних та обмежених у часі цілей. В контексті створення мобільного додатку, метод SMART може бути використаний для раціонального планування та управління процесом розробки з урахуванням конкретних потреб та очікувань стосовно функціональності та якості додатку. [10]

Метод SMART надає чітку структуру для визначення та формалізації мети проекту, а також допомагає уникнути загальних та неспецифічних формулювань цілей, що можуть призвести до недорозумінь та недосягнення бажаних результатів. Крім того, використання методу SMART дозволяє забезпечити вимірювання та оцінку прогресу розробки додатку, а також сприяє встановленню реалістичних термінів виконання завдань.

У контексті створення мобільного додатку, метод SMART може бути використаний для ефективного управління процесом розробки та досягнення поставлених цілей проекту. Використання методу SMART допомагає розробникам мобільних додатків уточнити та специфікувати свої цілі, забезпечуючи конкретність їх формулювання. Вимірюваність цілей дозволяє об'єктивно оцінювати прогрес роботи та досягнуті результати. Досяжність цілей допомагає уникнути постановки нереалістичних завдань, а релевантність дозволяє вирішувати проблеми, що є ключовими для успішного створення мобільного додатку. Крім того, визначення обмежень у часі сприяє збереженню графіка розробки та вчасному завершенню проекту.

Отже, можемо сформулювати мету нашого проекту за цими п’ятьма факторами. Результати наведені у таблиці 1.1.

Таблиця 1.1 – Формалізація мети за технологією SMART

| | |
|---|---|
| Specific | Інформаційне та програмне забезпечення клієнтської частини мобільного застосунку для ресторанного бізнесу |
| Measurable | Результат допоможе скоротити час (2 дні) та ресурси (кадрові та матеріальні) для створення зв’язку між клієнтом та додатком для ресторанного бізнесу. |
| Achievable | Розробка мобільного додатку здійснюється відповідно до рівня компетентності та на основі схваленого технічного завдання. |
| Revelant | Результат допоможе створити функціональну та ефективну клієнтську частину застосунку. |
| Time-bound | Реалізація проекту відбувається з дотриманням встановлених заздалегідь термінових обмежень, що були визначені на початковому етапі процесу у червні 2023 року. |

Work Breakdown Structure (WBS) - це ієрархічна декомпозиція проекту на окремі робочі елементи з метою більш ефективного планування, виконання та контролю проектних завдань. В контексті створення мобільного додатку, WBS може бути використаний для систематизації та організації всіх необхідних завдань та активностей, що включаються в процес розробки.

Структура WBS представляє собою деревоподібну декомпозицію проекту на послідовні рівні деталізації, починаючи від загальних фаз та етапів проекту і закінчуючи окремими завданнями або компонентами додатку. Кожен елемент WBS є однозначно визначеним та має конкретну роль у досягненні проектних цілей.

Застосування WBS в розробці мобільного додатку допомагає зрозуміти повну картину проекту, визначити всі потрібні компоненти та завдання, а також керувати ресурсами та строками виконання. Він сприяє систематизації процесу розробки, розподілу робочих завдань між командою, встановленню пріоритетів та забезпеченню контролю за прогресом робіт[11]. На рисунку Б.1 представлено WBS

проєкту щодо розробки клієнтської частини мобільного застосунку для ресторанного бізнесу.

## Планування структури виконавців

Одним з ефективних підходів до структурування проектів і розробки мобільних додатків є OBS (Organizational Breakdown Structure), що представляє собою ієрархічну структуру організаційних одиниць, в якій визначаються відповідальність, функції та зв'язки між різними частинами проекту. OBS забезпечує розподіл завдань, управління ресурсами та координацію дій усіх учасників проекту з метою досягнення успішних результатів.

Використання OBS структури при створенні мобільного додатку допомагає визначити ролі та відповідальність кожного з учасників проекту. Це забезпечує чіткість управління та комунікації, дозволяє ефективно розподілити робочі завдання та ресурси, а також покращує координацію роботи між різними командами, що працюють над розробкою додатку. Крім того, OBS структура допомагає виявити потенційні ризики та проблеми на ранніх етапах проекту, що сприяє їх вчасному вирішенню та запобігає затримкам у виконанні завдань [12].

На рисунку Б.2 представлено організаційну структуру планування проекту. Список виконавців, що функціонують в проекті описано в таблиці Б.2.

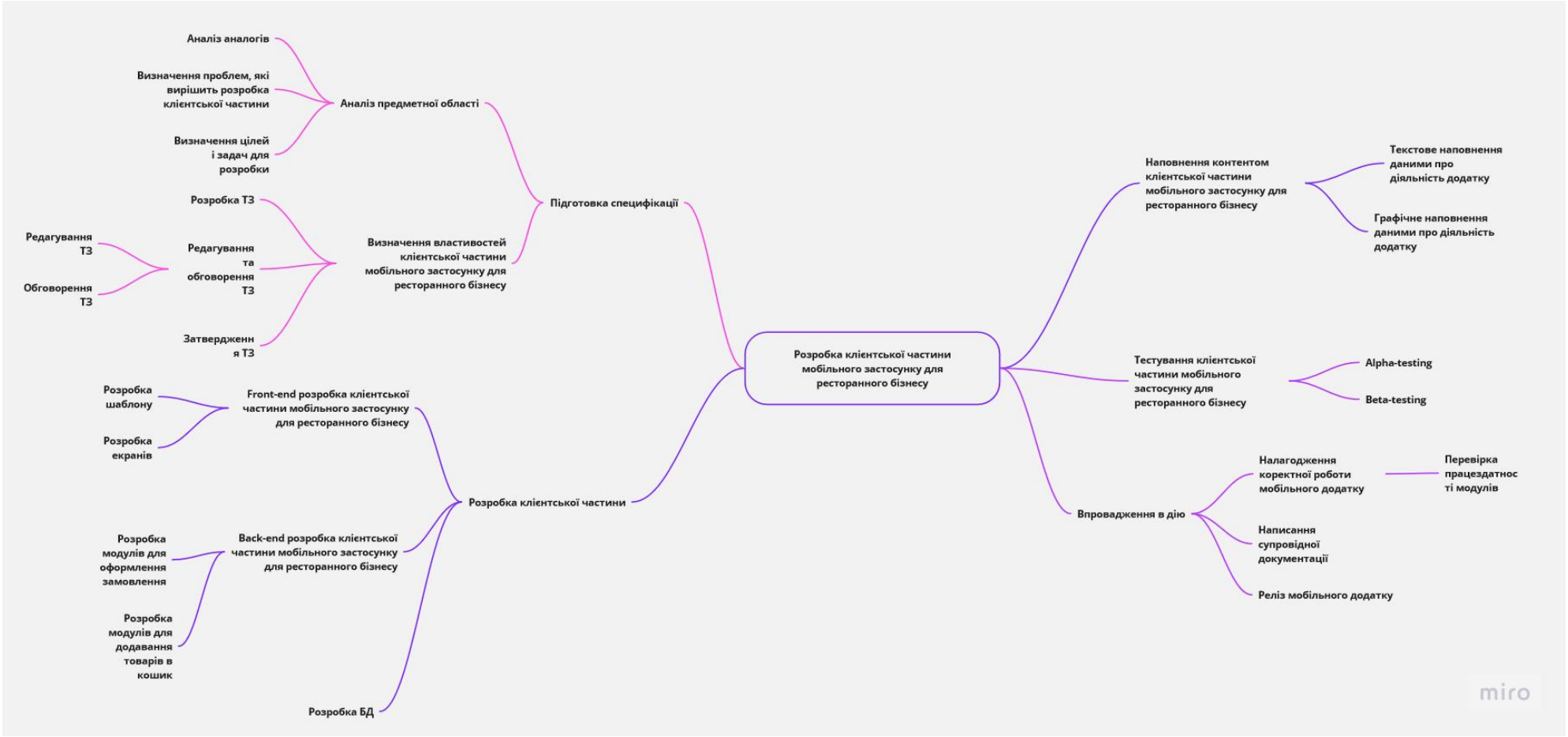


Рисунок Б.1 – WBS-структура робіт проекту

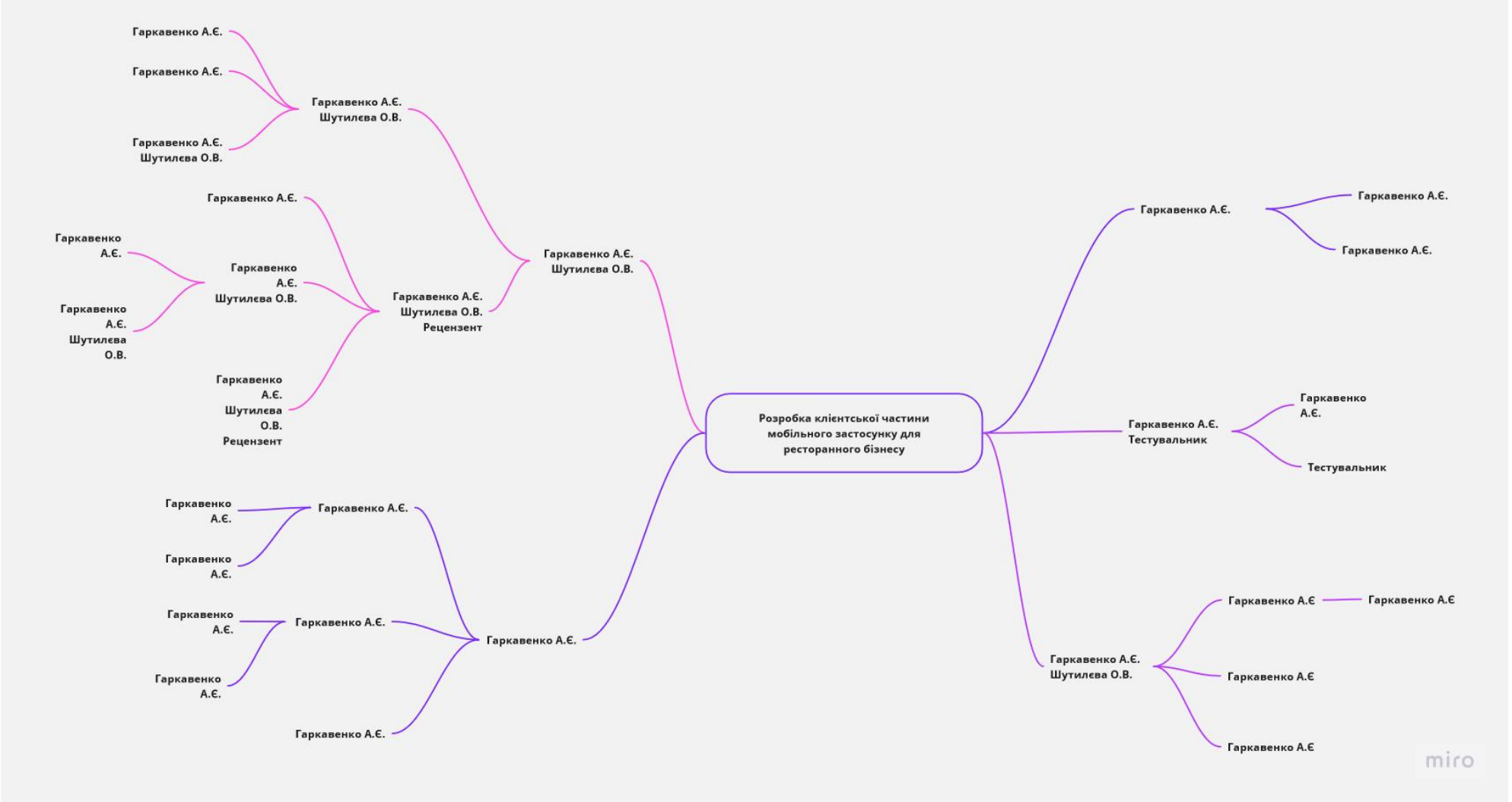


Рисунок Б.2 – OBS-структура робіт проект

Таблиця Б.2 – Виконавці проєкту

| Роль | Ім’я | Проєктна роль |
|---|---|---|
| Керівник проєкту | Шутилєва О.В. | Ця роль відповідає за дотримання графіку виконання завдань та ефективне розподілення ресурсів. Також керівник проєкту займається збором та аналізом даних. |
| Front-end-розробник | Гаркавенко А.Є. | Виконує front-end розробку мобільного додатку. |
| Back-end розробник | Гаркавенко А.Є. | Виконує back-end розробку мобільного додатку |
| Тестувальник | Тестувальник | Відповідає за тестування функціоналу мобільного додатку. |

## Діаграма Ганта

Діаграма Ганта є візуальним інструментом управління проектами, що дозволяє відображати послідовність та тривалість завдань у часовій шкалі. Вона використовується для планування, відстеження та керування проектами, включаючи створення мобільного додатку. Діаграма Ганта надає уявлення про загальний хід роботи над проектом, вказує на залежності між завданнями, та допомагає визначити критичні шляхи та резерви часу.

Календарний графік проєкту продемонстровано на рисунках Б.3-Б.4.

| | | Розробка клієнтської частини мобільного застосунку для ресторанного бізнесу | 93 дней | Ср 16.11.22 | Пт 24.03.23 | | |
|---|---|---|---|---|---|---|---|
| | | Ініціалізація проекту | 4 дней | Ср 16.11.22 | Пн 21.11.22 | | |
| | | Опрацювання ТЗ | 1 день | Ср 16.11.22 | Ср 16.11.22 | | Гаркавенко А.Є. |
| | | Аналіз контенту | 1 день | Ср 16.11.22 | Чт 17.11.22 | 3 | Гаркавенко А.Є. |
| | | Пошук контенту | 1 день | Чт 17.11.22 | Пт 18.11.22 | 4 | Гаркавенко А.Є. |
| | | Ідентифікація проекту | 1 день | Пт 18.11.22 | Пн 21.11.22 | 5 | Шутилєва О.В. |
| | | **Планування проекту** | **15 дней** | Пн 21.11.22 | Пн 12.12.22 | 6 | |
| | | Структура проекту | 2 дней | Пн 12.12.22 | Ср 14.12.22 | 7 | Гаркавенко А.Є. |
| | | Розробка дизайну та макету | 2 дней | Ср 14.12.22 | Пт 16.12.22 | 8 | Гаркавенко А.Є. |
| | | Розробка календарного плану «Розробка клієнтської частини мобільного застосунку для ресторанного бізнесу» | 5 дней | Пт 16.12.22 | Пт 23.12.22 | 9 | Гаркавенко А.Є. |
| | | Управління ресурсами | 1 день | Пт 23.12.22 | Пн 26.12.22 | 10 | Гаркавенко А.Є. |
| | | Управління фінансами | 1 день | Пн 26.12.22 | Вт 27.12.22 | 11 | Гаркавенко А.Є. |
| | | Управління якістю | 1 день | Вт 27.12.22 | Ср 28.12.22 | 12 | Гаркавенко А.Є. |
| | | Оптимізація плану | 1 день | Ср 28.12.22 | Чт 29.12.22 | 13 | Гаркавенко А.Є. |
| | | Затвердження плану | 1 день | Чт 29.12.22 | Пт 30.12.22 | 14 | Шутилєва О.В. |
| | | **Реалізація проекту** | 47 дней | Пн 02.01.23 | Вт 07.03.23 | | Гаркавенко А.Є. |
| | | **Розробка мобільного додатку** | 21 дней | Пн 02.01.23 | Пн 30.01.23 | | Гаркавенко А.Є. |

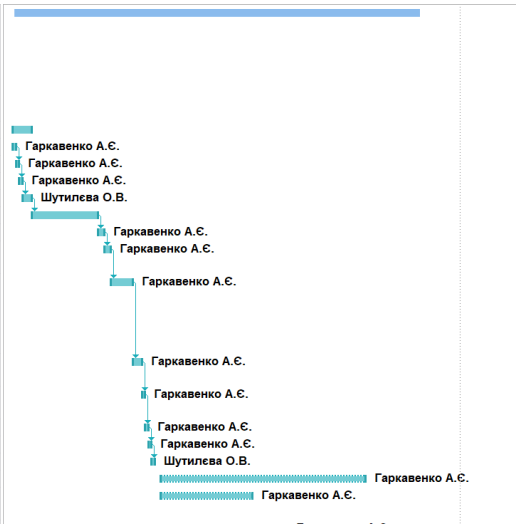


Рисунок Б.3 – Діаграма Ганта. Частина 1

| | | **Реалізація проекту** | 47 дней | Пн 02.01.23 | Вт 07.03.23 | | Гаркавенко А.Є. |
|---|---|---|---|---|---|---|---|
| | | **Розробка мобільного додатку** | 21 дней | Пн 02.01.23 | Пн 30.01.23 | | Гаркавенко А.Є. |
| | | Реалізація дизайну та макету | 9 дней | Вт 31.01.23 | Пт 10.02.23 | | Гаркавенко А.Є. |
| | | Верстка | 8 дней | Пт 10.02.23 | Ср 22.02.23 | 18 | Гаркавенко А.Є. |
| | | Розробка екранів | 5 дней | Ср 22.02.23 | Ср 01.03.23 | 19 | Гаркавенко А.Є. |
| | | Проектування БД | 3 дней | Ср 01.03.23 | Пн 06.03.23 | 20 | Гаркавенко А.Є. |
| | | Створення форм реєстрації | 1 день | Пн 06.03.23 | Вт 07.03.23 | 21 | Гаркавенко А.Є. |
| | | **Створення зв'язку між користувачем та додатком** | **21 дней** | Ср 08.03.23 | Ср 05.04.23 | | Гаркавенко А.Є. |
| | | Розробка модулю для додавання товару до замовлення | 1 день | Ср 08.03.23 | Ср 08.03.23 | | Гаркавенко А.Є. |
| | | Розробка модулю реєстрації нового користувача | 1 день | Ср 08.03.23 | Чт 09.03.23 | 24 | Гаркавенко А.Є. |
| | | Розробка модулю зміни паролю | 2 дней | Чт 09.03.23 | Пн 13.03.23 | 25 | Гаркавенко А.Є. |
| | | Реалізація зв'язку БД з додатком | 4 дней | Пн 13.03.23 | Пт 17.03.23 | 26 | Гаркавенко А.Є. |
| | | Тестування | 2 дней | Пт 17.03.23 | Вт 21.03.23 | 27 | Гаркавенко А.Є. |
| | | **Завершення проекту** | 6 дней | Вт 21.03.23 | Ср 29.03.23 | 28 | |
| | | Створення документації | 1 день | Ср 29.03.23 | Чт 30.03.23 | 29 | Гаркавенко А.Є. |
| | | Створення інструкції користувача | 2 дней | Пт 31.03.23 | Пн 03.04.23 | 30 | Гаркавенко А.Є. |
| | | Запуск і підтримка | 2 дней | Вт 04.04.23 | Ср 05.04.23 | 31 | Гаркавенко А.Є. |
| | | Архівація проекту | 1 день | Чт 06.04.23 | Чт 06.04.23 | 32 | Гаркавенко А.Є. |

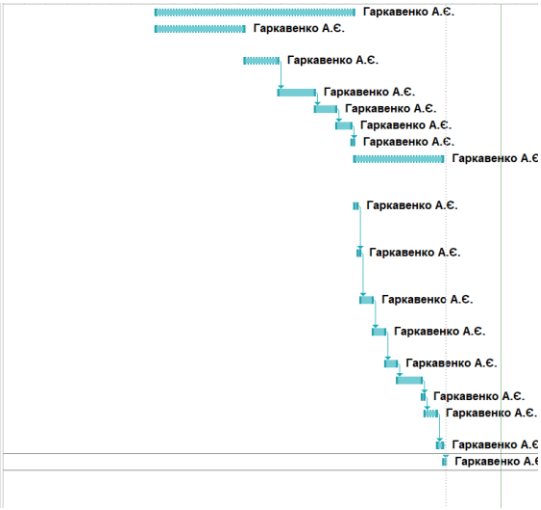


Рисунок Б.4 – Діаграма Ганта. Частина 2

## Управління ризиками проєкту

Управління ризиками проекту є процесом ідентифікації, аналізу, оцінки та керування ризиками, які можуть вплинути на успішне виконання проекту. Цей підхід спрямований на зменшення негативних наслідків ризиків та максимізацію можливостей в проекті. У контексті створення мобільного додатку, управління ризиками може забезпечити ефективну ідентифікацію та керування потенційними загрозами та можливостями, що виникають під час розробки та експлуатації додатку.

Управління ризиками проекту допомагає розробникам прогнозувати можливі проблеми та виявляти ключові області, які потребують уваги. Це дозволяє прийняти адекватні заходи перед виникненням проблем, зменшуючи ймовірність затримок у графіку, перевищення бюджету та інших негативних наслідків.

Для виконання якісної розробки мобільного додатку потрібно розглянути наступні ризики:

– Взаємні недорозуміння між розробником і замовником;

– Поява альтернативного додатку;

– Низька кваліфікація розробників;

– Неоптимальний розподіл часу;

– Оцінка та вибір неефективної технології розробки мобільного додатку;

– Неправильна оцінка в масштабі додатку;

– Помилки при розробці екранів;

– Несправності або неполадки в функціонуванні програмного забезпечення.;

– Відсутність резервних копій даних;

– Реалізація непотрібних скриптів;

– Невикористання моніторингу проекту;

– Дефекти або помилки, які виникли під час роботи програмного забезпечення;

– Зміна вимог замовника в процесі розробки додатку;

Таблиця Б.3 – Шкала оцінки ризику за вірогідністю виникнення та рівнем впливу.

| Оцінка | Вірогідність виникнення | Вплив ризику | Тип ризику |
|---|---|---|---|
| 1 | Низький | Низький | Допустимі |
| 2 | Середній | Середній | Виправдані |
| 3 | Високий | Високий | Неприпустимі |

Для зменшення негативного впливу ризиків на проект необхідно провести планування реагування на них. Цей процес включає визначення ефективності розробки та оцінку наслідків впливу ризиків на проект. Оцінка проводиться на основі показників, які описані в таблиці Б.3. В результаті планування реагування була отримана матриця ймовірності виникнення ризиків та впливу ризику, яка зображена на рисунку Б.4. Зеленим кольором позначаються прийнятні ризики, жовтим – виправдані, а червоним – неприпустимі.

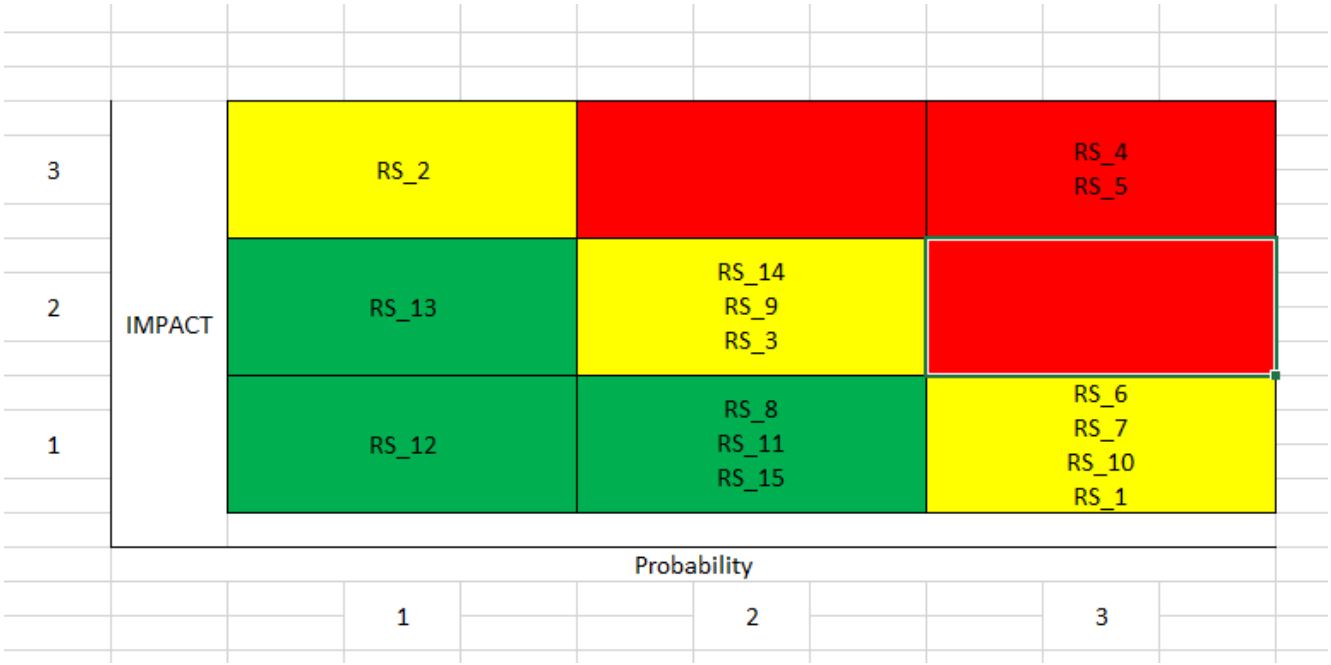


Рисунок Б.6. – Матриця ймовірності

Класифікація ризиків за рівнем, згідно з отриманим значенням індексу, продемонстрована у таблиці Б.4. У таблиці Б.5 описано ризики та стратегії реагування на кожен з них.

Таблиця Б.4 – Шкала оцінки за рівнем ризику.

| № | Назва | Межі | Перелік ризиків |
|---|---|---|---|
| 1 | Допустимі | 1<R<2 | 8,11,12,13,15 |
| 2 | Виправдані | 3<R<4 | 1,2,3,6,7,9,10,14 |
| 3 | Неприпустимі | 6<R<9 | 4,5 |

Таблиця Б.5 – Ризики та стратегії реагування

| ID | Статус ризику | Опис ризику | Вірогідність виникнення | Вплив ризику | Ранг ризику | План А | Тип стратегії реагування | План Б |
|---|---|---|---|---|---|---|---|---|
| RS_1 | Відкритий | Взаємні недорозуміння між розробником і замовником | Низька | Середній | 3 | 1. Забезпечити позитивні взаємини між розробником та керівником.<br>2. Ретельно дотримуватися вимог ділового спілкування.<br>3. Створити сприятливе середовище для ефективної співпраці. | Попередження | У разі виникнення непорозумінь необхідно одразу з'ясувати причину, що призвела до непорозуміння, обговорити її і забезпечити сприятливу атмосферу у колективі. |

Продовження табл. Б.5

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| RS_2 | Відкритий | Поява альтернативного додатку | Низька | Середній | 4 | 1. Здійснити передбачальний аналіз.<br>2. Визначити інноваційний підхід до розробки додатку. | Прийняття | Змінити концепцію розроблюваного додатку |
| RS_3 | Відкритий | Низька кваліфікація розробників | Середня | Середній | 4 | 1. Розширити професійні навички команди.<br>2. Залучити веб-ресурси для поліпшення рівня компетентності. | Пом’якшення | Заздалегідь прийняти до уваги необхідний час для підготовки спеціалістів шляхом надання корисних ресурсів та онлайн навчання. |

Продовження табл. Б.5

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| RS_4 | Відкритий | Не точне завдання на розробку | Середня | Високий | 6 | 1. Установити чіткі та зрозумілі замовнику вимоги під час детального обговорення.<br>2. Створити глосарій, що визначає та уточнює терміни та визначення, щоб уникнути неузгодженостей у розумінні термінів.<br>3. Проводити регулярний контроль проміжних етапів роботи замовником. | Попередження | У разі виявлення розбіжностей між характеристиками продукту та вимогами, необхідно детально і чітко визначити неправильно виконані аспекти та внести відповідні корективи. |

Продовження табл. Б.5

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| RS_5 | Відкритий | Неоптимальний розподіл часу | Висока | Високий | 9 | Виконати оцінку актуальності основних процесів та завдань, звернувши свою увагу на коректність розподілу часу. Розподілити виконання робіт за пріоритетом і дотримуватися календарного плану з точністю. | Пом’якшення | Переглянути порядок пріоритетів завдань і знайти можливості для оптимізації роботи з поточною структурою. Провести обговорення щодо змін термінів реалізації продукту з замовником(для внесення необхідних корекцій). |

Продовження табл. Б.5

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| RS_7 | Відкритий | Оцінка та вибір неефективної технології розробки мобільного додатку | Середня | Середній | 4 | 1. Провести детальний аналіз доступних методів та інструментів для реалізації додатку.<br>2. Вибрати просту і зрозумілу технологію розробки, яка легко використовується. | Пом'якшення | Призначити належний час та ресурси для проведення досліджень та пошуків можливих вдосконалень використовуваної технології.<br>Скористатися додатковими джерелами для отримання підтримки та допомоги. |

Продовження табл. Б.5

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| RS_8 | Відкритий | Неправильна оцінка в масштабі додатку | Низька | Середній | 2 | 1. Здійснити грунтовний аналіз проекту.<br>2. Встановити основні кроки реалізації та зробити розподіл часу.<br>3. Провести детальний аналіз масштабів проекту, користуючись допоміжними джерелами. | Пом’якшення | Перегляд масштабів проекту.<br><br>Переформулювання стратегії виконання проекту. |

Продовження табл. Б.5

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| RS_9 | Відкритий | Помилки при верстці екранів | Середня | Середній | 4 | Під час верстки важливо забезпечувати активну взаємодію замовника та систематично представляти проміжні результати для зворотного зв'язку. | Пом’якшення | Проводити періодичний моніторинг проміжних результатів під час реалізації проекту. |

Продовження табл. Б.5

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| RS_10 | Відкритий | Дефекти або помилки, які виникли під час роботи програмного забезпечення | Низька | Середній | 3 | 1. Забезпечити наявність додаткових програмних ресурсів для резервного використання.<br>2. Залучити експерта з відповідною кваліфікацією для вирішення проблем, пов'язаних з відмовами. | Попередження | Перехід на нове програмне забезпечення, яке є альтернативою до поточного рішення. |

Продовження табл. Б.5

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| RS _11 | Відкритий | Відсутність резервних копій даних | Ннизька | Середній | 2 | 1. Здійснити імплементацію автоматичного збереження даних.<br>2. Гарантувати збереження даних на різноманітних пристроях зберігання інформації. | Попередження | Здійснювати резервне копіювання даних після завершення кожного етапу. |

Продовження табл. Б.5

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| RS_12 | Відкритий | Реалізація непотрібних компонентів | Низька | Низький | 1 | Проінформувати замовника про можливу потребу у включенні додаткових компонентів. | Використання | Провести дискусію про вимоги та потенційні втрати, пов'язані зі змінами в проекті. |

Продовження табл. Б.5

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| RS_13 | Відкритий | Невикористання моніторингу проекту | Середня | Низький | 2 | Проводити регулярний моніторинг результатів проекту під час його виконання і забезпечувати постійний нагляд за прогресом проекту з боку працівників. | Перенесення | Замовник повинен здійснювати постійний контроль за проектом, включаючи моніторинг виконання та отримання проміжних результатів після кожного етапу. |

Продовження табл. Б.5

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| RS_14 | Відкритий | Виникнення проблем із програмним забезпеченням користувачів | Середня | Середня | 4 | 1. Планування розробки проекту з урахуванням вимог, що стосуються програмного забезпечення, які визначені користувачами проекту.<br>2. Адаптація проекту до різних версій програмного забезпечення, які будуть використовуватися, шляхом внесення необхідних змін. | Прийняття | |

Продовження табл. Б.5

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| RS_15 | Відкритий | Зміна вимог замовника в процесі розробки додатку | Низька | Середній | 3 | На початкових етапах необхідно досягти узгодження щодо всіх питань з метою зменшення кількості змін, необхідних у ході розробки. | Пом’якшення | Перегляд проекту є необхідним під час зміни вимог. |

# ДОДАТОК В

Лістинг коду inputform.js

```
import React, {FC, useContext, useRef} from 'react';
import {ErrorText, TextInputContainer} from './styled';

import {StyleSheet, TextInput, TouchableOpacity, View} from
'react-native';
import {TTextInputProps} from './types';
import {TextInputMask} from 'react-native-masked-text';
import {COLORS} from '@App/utils';
import SearchIcon from '@App/assets/icons/search.svg';
import CloseIcon from '@App/assets/icons/close.svg';
import {ThemeContext} from '@App/contexts';
import {TitleText} from '../text/TitleText';
import PhoneInput from 'react-native-phone-number-input';

const TextInputForm: FC<TTextInputProps> = ({
  value,
  onChange,
  errorText,
  inputContainerStyle = {},
  inputTextStyle = {},
  title,
  placeholder,
  isPhoneInput = false,
  keyboardType = 'default',
  editable = true,
  searchIcon = false,
  clearIcon = false,
  containerStyle = {
    marginVertical: 16,
```

```
  },
  autoComplete,
}) => {
  const {mode} = useContext(ThemeContext);
  const renderClearIcon = () => {
    if (clearIcon && value.length != 0) {
      return (
        <TouchableOpacity
          onPress={() => {
            onChange('');
          }}>
          <CloseIcon fill={COLORS[mode].icons.white} width={20}
height={20} />
        </TouchableOpacity>
      );
    }

    return null;
  };
  return (
    <View
      style={[
        containerStyle,
        {opacity: editable ? 1 : 0.7},
        isPhoneInput ? {width: '100%'} : {},
      ]}>
      {title && (
        <TitleText style={{marginLeft: 16, marginBottom: 10}}>
          {title}
        </TitleText>
      )}
```

```
      <TextInputContainer
        isPhoneInput={isPhoneInput}
        isError={!!errorText}
        style={{
          ...{inputContainerStyle},
          backgroundColor: COLORS[mode].inputBackground,
        }}>
        {searchIcon && <SearchIcon style={{marginRight: 10}} />}

        {isPhoneInput ? (
          <PhoneInput
            defaultValue={value}
            defaultCode="UA"
            layout="first"
            placeholder={placeholder}
            textInputProps={{
              maxLength: 9,
              placeholderTextColor: COLORS.gray,
            }}
            countryPickerButtonStyle={{
              display: 'none',
            }}
            textContainerStyle={{backgroundColor:
COLORS[mode].second}}
            codeTextStyle={{
              color: COLORS[mode].text.main,
              fontSize: 16,
              marginTop: -2,
            }}
            textInputStyle={{
              color: COLORS[mode].text.main,
              fontSize: 16,
```

```
            }}
            onChangeFormattedText={text => {
              onChange(text.replace('+', ''));
            }}
          />
        ) : (
          <TextInput
            value={value}
            onChangeText={onChange}
            editable={editable}
            placeholder={placeholder}
            placeholderTextColor={COLORS.gray}
            autoComplete={autoComplete}
            style={[
              style.defaultInputText,
              inputTextStyle,
              {color: COLORS[mode].text.main},
            ]}
            keyboardType={keyboardType ? keyboardType :
'default'}
          />
        )}

        {renderClearIcon()}
      </TextInputContainer>

      {!!errorText && <ErrorText>{errorText}</ErrorText>}
    </View>
  );
};

const style = StyleSheet.create({
```

```
  defaultInputText: {
    width: '100%',
    fontSize: 16,
  },
});
export default TextInputForm;
```

Лістинг коду productcard.js

```
import React, {FC, ReactNode, useCallback, useContext, useMemo}
from 'react';
import {
  TouchableContainer,
  OptionsContainer,
  ImageContainer,
  DeleteIcon,
  PriceText,
  SumText,
  Footer,
  Header,
  DescriptionContainer,
  WeightText,
  OldPriceText,
  TitleProductText,
} from './styled';

import {TProductCard} from './types';
import {Row} from '../styled';
import {styles} from './styles';
import {COLORS} from '../../utils';
import {FlatList, StyleSheet, View} from 'react-native';
import {useTranslation} from 'react-i18next';
import {useTypedSelector} from '@App/store/store';
```

```
import {useDispatch} from 'react-redux';
import {
  decrementProductLCAction,
  deleteAdditionsLCAction,
  deleteShoppingCardLocalAction,
  incrementProductLCAction,
} from '@App/store/shopCard';
import {PrintCorrectPriceCard, PrintCorrectPriceSV} from
'../PrintCorrectPrice';
import {Counter} from '../Counter';
import {AdditionCard} from '@App/screens/MenuStack/components';
import Trash from '@App/assets/icons/delete_ShopCard.svg';
import {ProductImage} from '@App/components/ProductImage';
import {ThemeContext} from '@App/contexts';
import {TitleText} from '../text/TitleText';

const ProductCard: FC<TProductCard> = ({
  mode,
  count,
  product,
  additions,
  variation,
  color,
}) => {
  const rest = useTypedSelector(s => s.rest.rest);

  const additionsPrice = useMemo(() => {
    return (
      additions
        .map(el => el.addition.price * el.count)
        .reduce((sum, cur) => sum + cur, 0) ?? 0
    );
```

```
  }, [additions]);

  const currentVariations = product.variations
    ? product.variations.find(el => el._id == variation)
    : undefined;

  const currentRestDiscount = useMemo(() => {
    return product.restaurantDiscount
      ? product.restaurantDiscount.map(el =>
el.restaurant).includes(rest?._id!)
      : false;
  }, [product.restaurantDiscount]);

  const {mode: modeTheme} = useContext(ThemeContext);

  const dispatch = useDispatch();

  const {t} = useTranslation();

  const onPressIncrementProduct = async (id: string) => {
    dispatch(
      incrementProductLCAction({
        id,
        addition: additions
          .map(el => el._id)
          .sort()
          .join(),
        variation: variation,
      }),
    );
  };
```

```
const onPressDecrementProduct = async (id: string) => {
  if (count > 1) {
    dispatch(
      decrementProductLCAction({
        id,
        addition: additions
          .map(el => el._id)
          .sort()
          .join(),
        variation: variation,
      }),
    );
  }
};

const onDeleteInCard = (id: string) => {
  dispatch(
    deleteShoppingCardLocalAction({
      id,
      addition: additions
        .map(el => el._id)
        .sort()
        .join(),
      variation: variation,
    }),
  );
};

const onDeleteAddition = (id: string) => {
  dispatch(
    deleteAdditionsLCAction({
      id: product._id,
```

```
        addition: additions
          .map(el => el._id)
          .sort()
          .join(),
        variation: variation,
        additionsID: id,
      }),
    );
  };

  const renderHeader = useCallback<() => ReactNode>(() => {
    return (
      <Header mode={modeTheme} style={color ? {backgroundColor:
color} : {}}>
        {mode == 'card' && (
          <DeleteIcon
            activeOpacity={0.8}
            onPress={() => onDeleteInCard(product._id)}>
            <Trash
              width={24}
              height={24}
              fill={COLORS[modeTheme].icons.white}
              style={{margin: 4}}
            />
          </DeleteIcon>
        )}

        <DescriptionContainer>
          <ImageContainer>
            <ProductImage
              photo={product.photo}
              style={styles.image}
```

```
              styleDefImg={{width: '100%', height: '100%'}}
            />
          </ImageContainer>
<View style={styles2.descContainer}>
            <TitleText
              style={{width: '90%'}}
              numberOfLines={2}
              lineBreakMode="tail">
              {product.name}{' '}
              {currentVariations ? (${currentVariations.name}) :
''}
            </TitleText>

            <View style={styles2.priceContainer}>
              {!!product.weight &&
<WeightText>{weight}</WeightText>}

              <View style={{flexDirection: 'row'}}>
                {currentRestDiscount && (
                  <OldPriceText>
                    {(
                      product.price +
(currentVariations?.priceChange ?? 0)
                    ).toString()}
                  </OldPriceText>
                )}

                <PriceText numberOfLines={1}
lineBreakMode="tail">
                  {PrintCorrectPriceSV({
                    price: product.price,
                    restId: rest?._id!,
                    stockPrice: product.stockPrice,
```

```
                    restaurantDiscount: currentRestDiscount,
                    selectedVariant: currentVariations,
                    additionsPrice: 0,
                  })}{' '}
                  {t('components.currency')}
                </PriceText>
              </View>
            </View>
          </View>
        </DescriptionContainer>
      </Header>
    );
  }, [product, modeTheme]);

  const renderOptions = useCallback<() => ReactNode>(() => {
    return (
      <OptionsContainer isOneLine={additions.length == 0}
mode={modeTheme}>
        <FlatList
          data={additions}
          keyExtractor={el => el._id}
          showsVerticalScrollIndicator={false}
          bounces={false}
          scrollEnabled={false}
          style={{width: '100%'}}
          renderItem={({item}) => {
            return (
              <AdditionCard
                counterMode={mode == 'card' ? 'counter' :
'price'}
                {...item.addition}
                onPressAdds={count =>
onDeleteAddition(item.addition._id)}
```

```
                count={1}
              />
            );
          }}
        />
      </OptionsContainer>
    );
  }, [product, additions, modeTheme]);

  const renderSumBlock = useCallback<() => ReactNode>(() => {
    return (
      <Row justify="space-between" style={styles.sumRow}>
        {mode == 'card' ? (
          <Counter
            count={count || 0}
            onMinus={count =>
onPressDecrementProduct(product._id)}
            onPlus={count =>
onPressIncrementProduct(product._id)}
          />
        ) : (
          <WeightText>
            {t('singleOrder.amount')} {count}
          </WeightText>
        )}

        <SumText
          numberOfLines={1}
          lineBreakMode="tail"
          style={{
            textAlign: 'right',
          }}>
```

```
          {PrintCorrectPriceCard({
            price: product.price,
            restId: rest?._id!,
            stockPrice: product.stockPrice,
            restaurantDiscount: currentRestDiscount,
            variation: currentVariations,
            additionsPrice: additionsPrice,
            count: count,
          })}{' '}
          {t('components.currency')}
        </SumText>
      </Row>
    );
  }, [
    count,
    product,
    additionsPrice,
    currentVariations,
    product.stockPrice,
    modeTheme,
  ]);

  const weight = useMemo(
    () =>
      product.weight
        ? product.variations.find(el => el._id ==
variation)?.weightChange
          ? product.variations.find(el => el._id ==
variation).weightChange
          : product.weight
        : '0',
    [product.weight],
```

```
  );

  return (
    <TouchableContainer
      mode={modeTheme}
      style={styles.container}
      activeOpacity={1}>
      {renderHeader()}

      <Footer mode={modeTheme} style={color ? {backgroundColor:
color} : {}}>
        {renderOptions()}

        {renderSumBlock()}
      </Footer>
    </TouchableContainer>
  );
};
const styles2 = StyleSheet.create({
  descContainer: {
    flex: 1,
    flexDirection: 'column',
    justifyContent: 'space-between',
  },
  priceContainer: {
    marginTop: 10,
    flexDirection: 'row',
    justifyContent: 'space-between',
  },
});

export default ProductCard;
```