МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра інформаційних технологій

«До захисту допущено»
В.о. завідувача кафедри

_____ Світлана ВАЩЕНКО

____ _____ 2024 р.

КВАЛІФІКАЦІЙНА РОБОТА
на здобуття освітнього ступеня бакалавр

зі спеціальності 122 «Комп'ютерні науки»,

освітньо-професійної програми «Інформаційні технології проєктування»

на тему: «Мобільний додаток підтримки діяльності кав'ярні»

Здобувача групи ІТ-02 Чалий Нікіта Сергійович
(шифр групи) (прізвище, ім'я, по батькові)

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

_____ Нікіти ЧАЛОГО
(підпис) (Ім'я та ПРІЗВИЩЕ здобувача)

Керівник к. т. н., доц. Вікторія АНТИПЕНКО _____
(посада, науковий ступінь, вчене звання, Ім'я та ПРІЗВИЩЕ) (підпис)

Суми – 2024

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

ЗАТВЕРДЖУЮ
В. о. зав. кафедри ІТ

_____Світлана ВАЩЕНКО
«__» _____ 2024 р.

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ БАКАЛАВРА СТУДЕНТУ

Чалий Нікіта Сергійович

1. Тема роботи *Мобільний додаток підтримки діяльності кав'ярні*

керівник роботи *Антипенко Вікторія Петрівна, к.т.н., доцент*,

затверджені наказом по університету від «*8*» квітня 2024 р. №0588-VI

2. Строк подання студентом роботи «26» травня 2024 р.

3. Вхідні дані до роботи технічне завдання на розробку мобільного додатку підтримки діяльності кав'ярні

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) вступ, аналіз предметної області, проєктування мобільного додатку, розробка мобільного додатку, висновки

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) актуальність, постановка задачі, аналіз продуктів-аналогів, порівняння продуктів-аналогів, функціональні вимоги, функціональне моделювання з точки зору користувача, діаграма варіантів використання, архітектура мобільного додатку, практична реалізація мобільного додатку, демонтрація роботи мобільного додатку, висновки

6. Консультанти розділів роботи:

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7.Дата видачі завдання «04» квітня 2024 р

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Оформлення планування робіт	05.04.2024 12.04.2024	
2	Оформлення технічного завдання	12.04.2024 19.04.2024	
3	Проведення аналізу предметної області	19.04.2024 26.04.2024	
4	Проведення проєктування мобільного додатку	26.04.2024 03.05.2024	
5	Розробка та тестування мобільного додатку	03.05.2024 17.05.2024	
6	Підготовка мобільного додатку до реалізації	17.05.2024 20.05.2024	
7	Оформлення пояснювальної записки	20.05.2024 25.05.2024	

Студент _____ Нікіти ЧАЛОГО

(підпис)

Керівник роботи _____ к.т.н., доц. Вікторія АНТИПЕНКО

(підпис)

АНОТАЦІЯ

Тема кваліфікаційної роботи бакалавра: «Мобільний додаток підтримки діяльності кав’ярні».

Пояснювальна записка складається зі вступу, 3 розділів, висновків, списку використаних джерел із 37 найменувань, додатків. Загальний обсяг роботи − 75 сторінок, у тому числі 37 сторінок основного тексту, 4 сторінки списку використаних джерел, 34 сторінок додатків.

Актуальність роботи полягає в необхідності створення ефективного та зручного інструменту підтримки діяльності кав’ярні у вигляді відповідного мобільного додатку. Його використання допоможе закладу удосконалити виконання таких процесів, як оформлення замовлення, управління персоналом тощо за рахунок їх автоматизації. А також дозволить покращити комунікацію з клієнтами та рівень їх обслуговування.

Метою даної роботи є розробка кросплатформенного мобільного додатку підтримки діяльності кав’ярні. Для реалізації проєкту вирішено використовувати такі сучасні технології, як React Native для створення кросплатформених додатків, Tailwind CSS для стилізації, Node.js та Express.js для серверної частини, Firebase для управління даними та Postgresql для ефективного обміну даними.

Розроблений мобільний додаток може бути впроваджений у роботу кав’ярні для покращення обслуговування клієнтів, управління замовленнями та удосконалення виконання внутрішніх процесів. Це дозволить належним чином організувати підтримку діяльності такого закладу. Результати даного проєкту можуть бути використані для подальшого розвитку мобільного додатку та впровадження нових функцій.

Ключові слова: МОБІЛЬНИЙ ДОДАТОК, КАВ’ЯРНЯ, REACT NATIVE, TAILWIND CSS, NODE.JS, EXPRESS.JS, POSTGRESQL, GRAPHQL, КРОСПЛАТФОРМЕННІСТЬ.

ЗМІСТ

ВСТУП

Розвиток інформаційних технологій (ІТ) і поширення мобільних пристроїв останнім часом значно змінили уявлення про ведення бізнесу. Завдяки мобільним додаткам, клієнти мають можливість зручно та швидко замовляти страви та напої, резервувати столи та отримувати актуальну інформацію про події та акції у відповідних закладах. Одними із таких є кав'ярні. Сьогодні вони активно використовують сучасні програмні продукти для підтримки власної діяльності.

У цьому контексті створення мобільного додатку підтримки діяльності кав'ярні стає актуальною та перспективною задачею. Застосування даної розробки має своє переваги. Це не лише спростить процес оформлення замовлення для клієнтів у відповідному закладі продажу кавових напоїв, але й дозволить власникам бізнесу вдосконалити процес управління кав'ярнею та забезпечить більш ефективний обмін інформацією зі своїми відвідувачами.

Мета даної роботи полягає у розробці кросплатформенного мобільного додатку підтримки діяльності кав'ярні. Його застосування дозволить закладу удосконалити виконання певних процесів. Це оформлення замовлень гостями закладу, управління персоналом тощо за рахунок їх автоматизації. У свою чергу також покращиться комунікація з клієнтами та підвищиться рівень їх обслуговування.

Для досягнення поставленої мети необхідно виконати наступні задачі:

– провести детальний аналіз предметної області та огляд останніх публікацій по розробці мобільних додатків підтримки діяльності сучасного бізнесу;

– визначити актуальність роботи та цільову аудиторію створюваного мобільного додатку;

– провести аналіз існуючих продуктів-аналогів, виділити їх переваги та недоліки;

– обрати технології для розробки кросплатформенного мобільного додатку підтримки діяльності кав’ярні;

– виконати структурно-функціональне моделювання даного мобільного додатку;

– реалізувати структуру, функціональні можливості та інтерфейс відповідного мобільного додатку;

– виконати інтеграцію з базою даних;

– провести тестування розробленого мобільного додатку підтримки діяльності кав’ярні.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд останніх досліджень і публікацій

Мобільний додаток є програмним забезпеченням [1], спеціально розробленим для використання на мобільних пристроях. Наприклад, таких, як смартфони та планшети. Даний вид програм став необхідним із введенням сучасних мобільних технологій. Це надає користувачам зручний спосіб отримання доступу до різноманітних сервісів тощо.

Мобільні додатки поділяються на декілька типів у залежності від їхнього призначення та наявних функцій. Перший – це нативні додатки [2]. Вони розробляються спеціально для конкретної операційної системи (ОС), такої як iOS [3] чи Android [4]. Нативні додатки взаємодіють із ОС і використовують її можливості максимально ефективно, забезпечуючи високий рівень продуктивності та користувацького досвіду.

Другий типом виступають вебдодатки [5]. Вони базуються на технологіях веброзробки та запускаються в мобільному браузері. Такі вебдодатки не вимагають завантаження з магазинів додатків та можуть працювати на будь-якій операційній системі. Їх зазвичай легше оновлювати, оскільки зміни відбуваються на сервері, а не на самому пристрої.

Третій тип – гібридні додатки [6]. Вони поєднують у собі як елементи нативних, так і вебдодатків. Гібридні додатки написані мовами веброзробки, такими як HTML, CSS та JavaScript [7]. Вони зазначаються як «обгортки» для доступу до функціональності операційної системи. Це дозволяє поєднувати переваги нативних та вебдодатків.

Важливим аспектом розробки мобільних додатків є адаптація до особливостей різних операційних систем. iOS та Android, як операційні системи, мають свої унікальні стандарти дизайну та використовують різні мови програмування: Swift та Objective-C [8] відповідно для iOS, а Java та Kotlin[9] — для Android. Розробка нативних мобільних додатків зазвичай включає створення

окремих версій для кожної з цих платформ, що може потребувати окремих процесів розробки для кожної з них, однак за допомогою React Native [10] можна створити єдиний кодовий базис, який буде працювати на обох платформах, значно скорочуючи час на розробку. Це дає можливість використовувати ті ж та React-компоненти, які забезпечують нативний вигляд та на Android та iOS.

Окрім того, аспекти безпеки та конфіденційності даних є важливими при створенні мобільних додатків. Користувачі зазвичай обмінюються особистою інформацією та здійснюють онлайн-платежі через власні смартфони. Також, забезпечення швидкої та надійної роботи на різних пристроях важливе для забезпечення позитивного користувацького досвіду.

На основі всього вищезазначеного можна зробити висновок, що розробка мобільного додатку, який підтримує високий рівень продуктивності, безпеки та користувацького досвіду на різних операційних системах, є надзвичайно актуальною сьогодні.

1.2 Аналіз програмних продуктів-аналогів

Розгляд аналогів атку є досить важливим. Дослідження їхніх особливостей, функціоналу та якість забезпечення користувальницького досвіду відіграють вагому роль при реалізації власного проєкту. Аналіз аналогів надасть цінну інформацію про галузеві стандарти, очікування клієнтів та успішні стратегії, які можна використати в процесі розробки.

Одним із відомих відповідних аналогів є мобільний додаток Starbucks [11] (рис. 1.1). Він став еталоном для наступних розробок, орієнтованих на підтримку діяльності кафе та кав’ярень. Додаток Starbucks пропонує широкий спектр функцій, створених для покращення клієнтського досвіду, включаючи онлайн замовлення, винагороди за лояльність, персоналізовані рекомендації та платіжну інтеграцію.

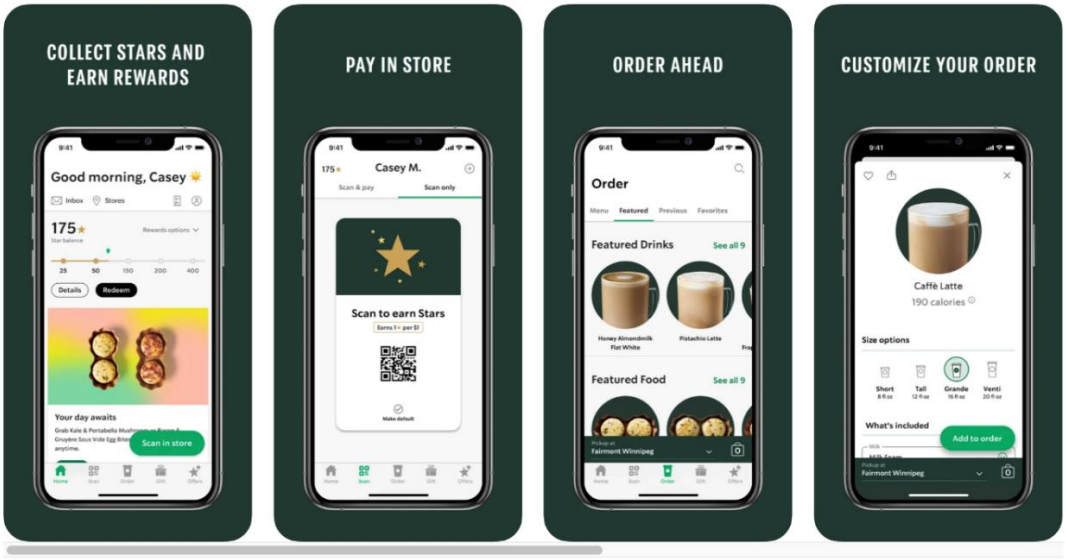

Рисунок 1.1 – Мобільний додаток Starbucks

Вивчаючи мобільний додаток Starbucks, можно отримати уявлення про ефективні стратегії для оптимізації процесу замовлення, впровадження програм лояльності та безперешкодної інтеграції платіжних рішень. Однак його користувачі часто мають проблеми, пов’язані з затримками у виконанні замовлень та недоліками у відображенні актуальної інформації про наявність товарів.

Іншим важливим аналогом є Dunkin [12] (рис. 1.2). (раніше Dunkin’ Donuts). Він є досить надійним мобільний додатком для організації обслуговування клієнтів кафе. Подібно до розробки Starbucks, Dunkin’ надає пріоритет зручності та персоналізації, дозволяючи користувачам робити замовлення заздалегідь, налаштовувати свої напої, отримувати винагороди та здійснювати платежі.

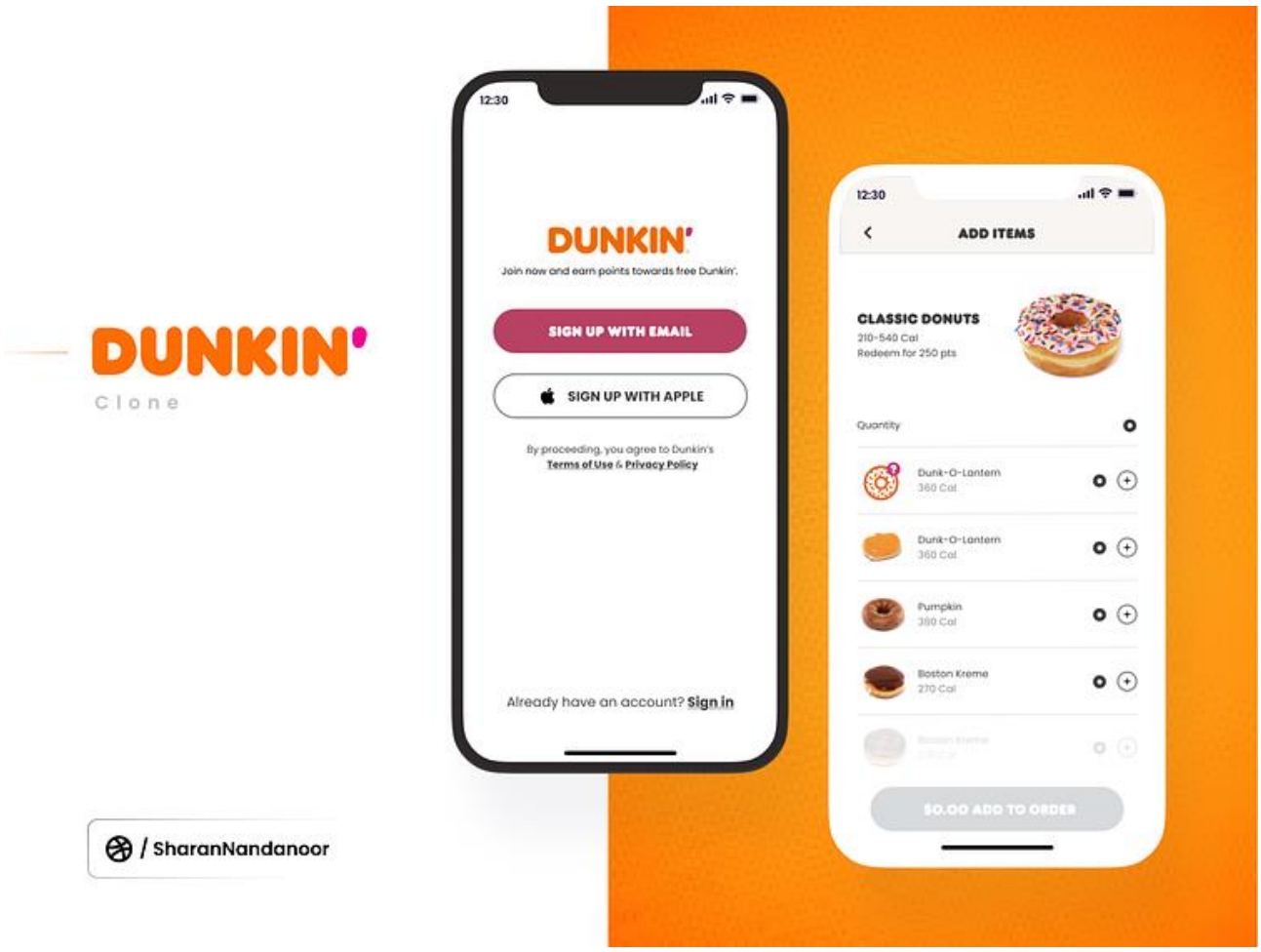

Рисунок 1.2 – Мобільний додаток Dunkin

Аналіз мобільного додатку Dunkin’ надає цінну інформацію щодо оптимізації користувацького інтерфейсу, розширення можливостей кастомізації замовлень та ефективного впровадження програм лояльності. Проте його клієнти іноді мають проблеми з навігацією та швидкістю роботи. Також часом недостатньо точно відображаються акції та знижки, а саме не завжди актуальна інформація.

Крім того, було проаналізовано мобільний додаток для доставки Grubhub [13] (рис. 1.3). Цей аналог дещо відрізняється від попередніх. Його застосування дає уявлення про те, як безперешкодно інтегрувати служби доставки у мобільний додаток підтримки діяльності кафе. Додаток Grubhub пропонує користувачам широкий вибір ресторанів, забезпечує прозорість ціноутворення, дозволяє відстежувати замовлення в режимі реального часу та надає зручні способи оплати.

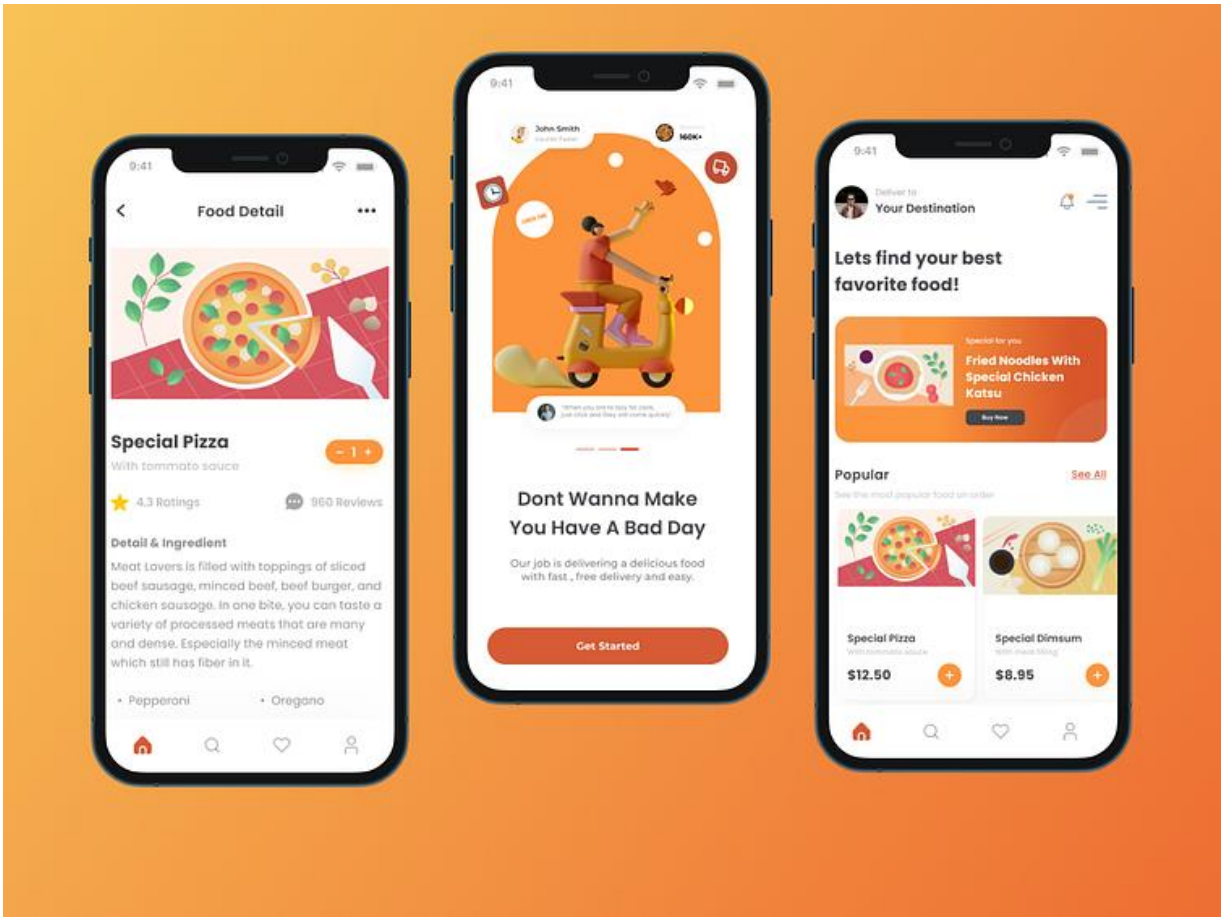

Рисунок 1.3 – Мобільний додаток Grubhub

Мобільний додаток Grubhub, хоча й надає зручний спосіб замовлення їжі, його клієнти часто мають проблеми, пов’язані з затримками в доставці та нестабільною його роботою. Це може викликати незручності для користувачів.

Результати проведеного аналізу аналогів мобільного додатку підтримки діяльності кав’ярні надані а таблиці 1.1.

Таблиця 1.1 – Порівняльна таблиця продуктів-аналогів

Параметри/Мобільні додатки	**Starbucks**	**Dunkin**	**Grubhub**	**Власний додаток**
Зручність	+	+/-	+	+
Доступність	+	+	+	+
Швидкодія	+/-	+	+	+
Підтримка	+	+/-	+	+
Кросплатформеність	+	+	+/-	+
Можливість завантажити з App store /Play market [14]	+	+	+	+

Дані з таблиці 1.1 показують, які ключові елементи повинен мати мобільний додаток підтримки діяльності кав'ярні належної якості. Основна увага має бути зосереджена на розробці інтуїтивно зрозумілого інтерфейсу та високої швидкодії. Усе це є ваговим для забезпечення зручності та високої якості обслуговування користувачів. Також важливо подолати виявлені недоліки.

Порівняння вищезазначених продуктів-аналогів підкреслює необхідність наявності як привабливого дизайну, так і можливості персоналізації замовлень у власній розробці. Також важливою є постійна підтримка користувачів. А мінімізація можливих помилок у замовленнях є одним із вирішальних аспектів для успішності створюваного програмного продукту на ринку. Це дозволить залучати нових клієнтів і підтримувати їхній інтерес до використання відповідного мобільного додатку.

1.3 Мета та задачі дослідження

Мета даного проєкту – розробка кросплатформенного мобільного додатку підтримки діяльності кав'ярні. Функціонал має бути мінімальним, але в той же час задовольняти всі потреби користувачів. Даний мобільний додаток повинен забезпечити належну роботу системи замовлень, управління персоналом і звітністю. Це надасть адміністраторам можливість удосконалити виконання бізнес-процесів відповідної кав'ярні за рахунок їх автоматизації.

Для досягнення поставленої мета даного проєкту треба виконати такі задачі:

– провести детальний аналіз предметної області та огляд останніх публікацій по розробці мобільних додатків підтримки діяльності сучасного бізнесу;

– визначити актуальність роботи та цільову аудиторію створюваного

мобільного додатку;

– провести аналіз існуючих продуктів-аналогів, виділити їх переваги та недоліки;

– обрати технології для розробки кросплатформенного мобільного додатку підтримки діяльності кав’ярні;

– виконати структурно-функціональне моделювання даного мобільного додатку;

реалізувати структуру, функціональні можливості та інтерфейс відповідного мобільного додатку (back-end і front-end частини [15-16]);

– виконати інтеграцію з базою даних;

– провести тестування розробленого мобільного додатку підтримки діяльності кав’ярні.

Даний мобільний додаток вирішено реалізовувати з використанням React Native [17] для front-end частини та Tailwind CSS [18] (за допомогою бібліотеки react-native-tailwindcss) для стилізації. Для серверної частини буде застосовано Node.js [19] та Express.js, [20]а для запуску на смартфоні – платформу Expo. Firebase та GraphQL [21] будуть використовуватись для інтеграції з сервером та управління даними. Уся інформація буде зберігатися у PosgresSql. Це забезпечить легкість у веденні звітності та управлінні замовленнями. Робота з даними буде організована належним чином через гнучкі запити GraphQL. Розробка буде супроводжуватися контролем версій за допомогою GIT [22].

Основні вимоги до створення мобільного додатку підтримки діяльності кав’ярні наступні:

– реалізація підтримки роботи на різноманітних девайсах для зручності всіх користувачів;

– забезпечення гнучкості у налаштуваннях, що дозволить адаптувати даний мобільний додаток під конкретні потреби кав’ярні, включаючи кастомізацію меню;

– реалізація інтеграції з існуючими pos-системами для удосконалення процесу управління замовленнями та звітністю;

– забезпечення підтримки програм лояльності та персоналізованих рекомендацій для заохочення клієнтів;

– реалізація можливості зворотного зв’язку для підтримки користувачів, щоб швидко реагувати на їх запити та вирішувати проблеми.

У Додатку А представлено технічне завдання, в якому описані вимоги на розробку даного проєкту. Детальне планування його робіт представлено у Додатку Б.

2 ПРОЄКТУВАННЯ МОБІЛЬНОГО ДОДАТКУ

2.1 Структурно-функціональне модулювання

Для ілюстрації функціональності та роботи запропонованого мобільного додатку підтримки діяльності кав'ярні було створено діаграму в нотації DEF0 [23]. Вона складається з графічного опису процесів, які виконує програмний продукт, і відображає їх взаємозв'язки. Створення діаграми IDEF0 допомагає краще зрозуміти функції мобільного додатку та його компоненти.

На рисунку 2.1. представлена контекстна діаграма IDEF0 для запропонованого мобільного додатку підтримки діяльності кав'ярні. На ній зображено функції даного програмного продукту та його взаємодію з різними компонентами. Основним процесом є «Оформлення замовлення кави».

Вхідними даними є «Запит на замовлення кави». Він надходить від користувачів через мобільний додаток. Цей запит зберігається в базі даних. Таким чином мобільний додаток слугує інтерфейсом для замовлення товару в кав'ярні. Також вхідними даними є «Контакні дані користувача» та «Платіжні дані користувача». Вони потрібні на етапах формування та оплати замовлення.

Обмеженнями виконання вищезазначеного процесу є асортимент кав'ярні та правила оформлення замовлення. Це необхідно для регулювання роботи даного мобільного додатку.

Механізмами є мобільний пристрій, мобільний додаток, користувач, база даних і платіжна система.

Вихідними даними є «Сформоване замовлення». Це результат обробки та виконання замовлень, що надходять від користувачів.

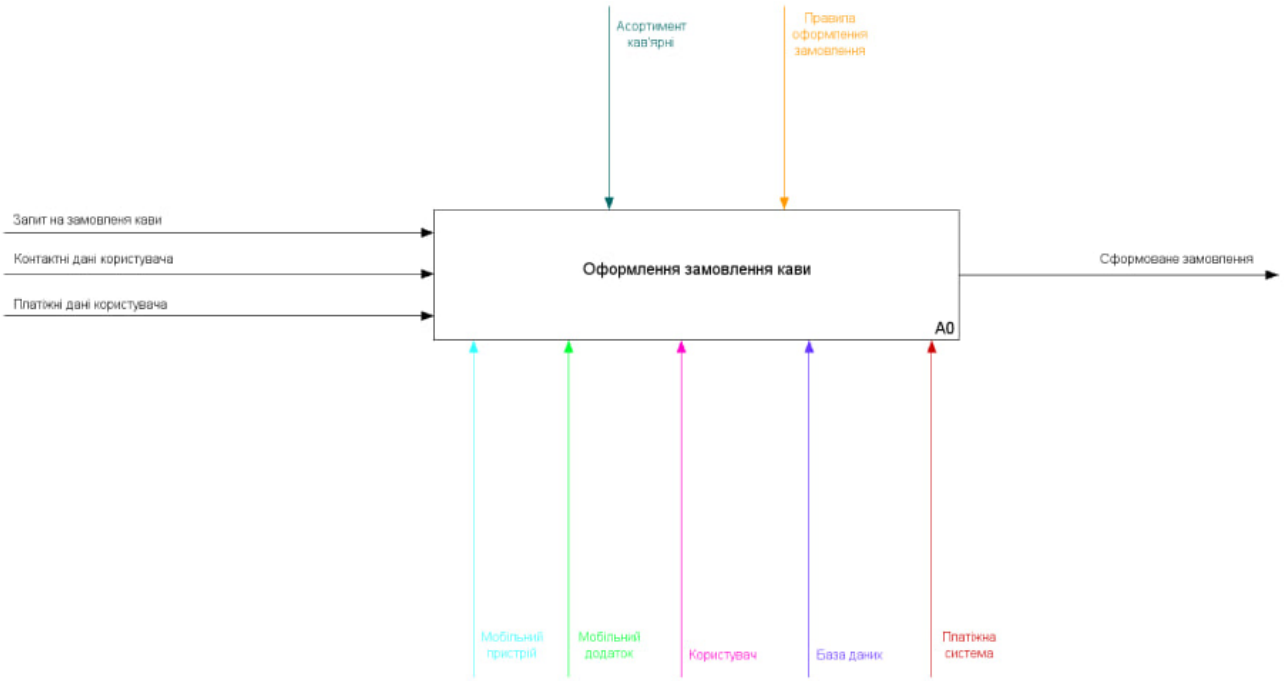

Рисунок 2.1 – Контекстна діаграма IDEF0

На рисунку 2.2 наведена декомпозиція контекстної діаграми IDEF0 [24]. Вона показує більш деталізовану структуру та взаємозв’язки компонентів мобільного додатку підтримки діяльності кав’ярні.

Ця декомпозиція включає наступні елементи:

– підсистеми та процеси, які складаються з набору функціональних блоків та їх зв’язків;

– функціональні блоки, які відповідають конкретним завданням або операціям, що виконуються в межах даного програмного продукту;

– зв’язки між функціональними блоками, які показують потоки даних або керування між ними.

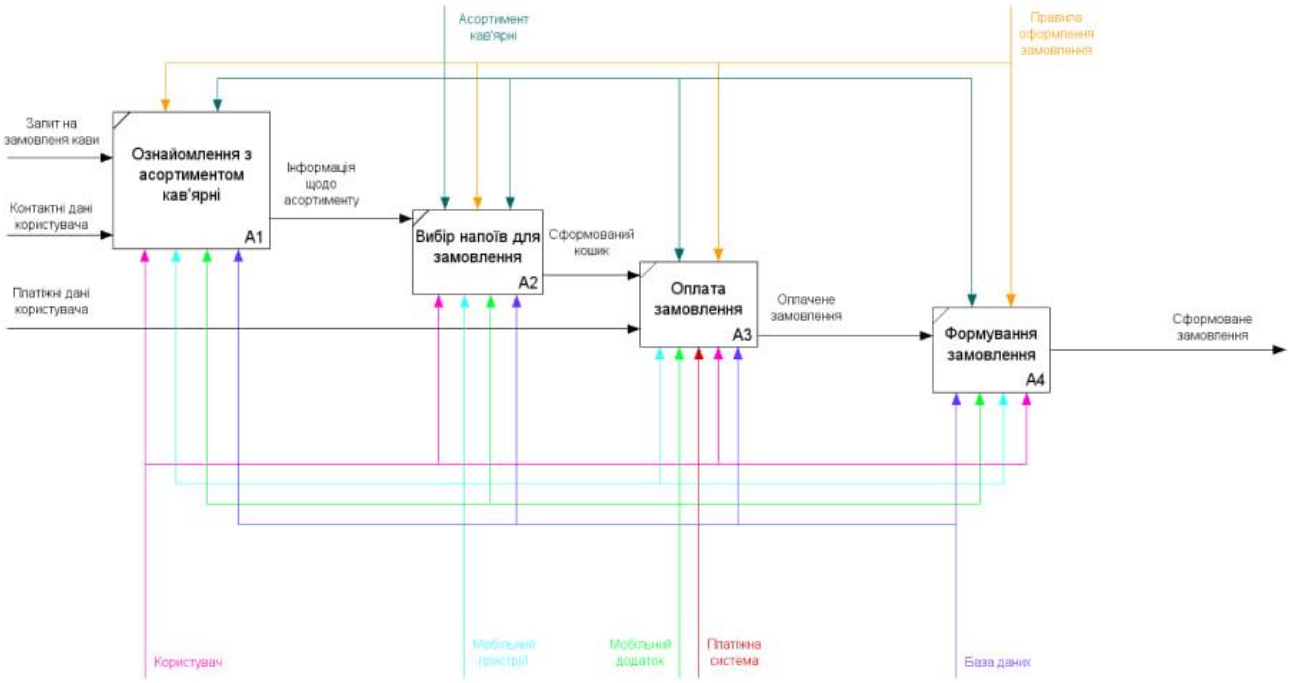

Рисунок 2.2 – Декомпозиція контекстної діаграми IDEF0

Ця декомпозиція дозволяє розглянути представлений мобільний додаток на більш глибокому рівні, розкриваючи його складові частини та взаємодію між ними. Вона допомагає розробникам краще зрозуміти функціональність програмного продукту та виявити можливі проблеми або покращення, які можуть бути внесені до проєкту.

2.2 Моделювання варіантів використання

Діаграма варіантів використання [25] є важливим елементом при проєктуванні програмного продукту. Вона показує взаємозв’язки між ним і акторами. А також відображає функціональні вимоги зі сторони користувачів. На рисунку 2.3 зображена діаграма варіантів використання мобільного додатку підтримки діяльності кав’ярні.

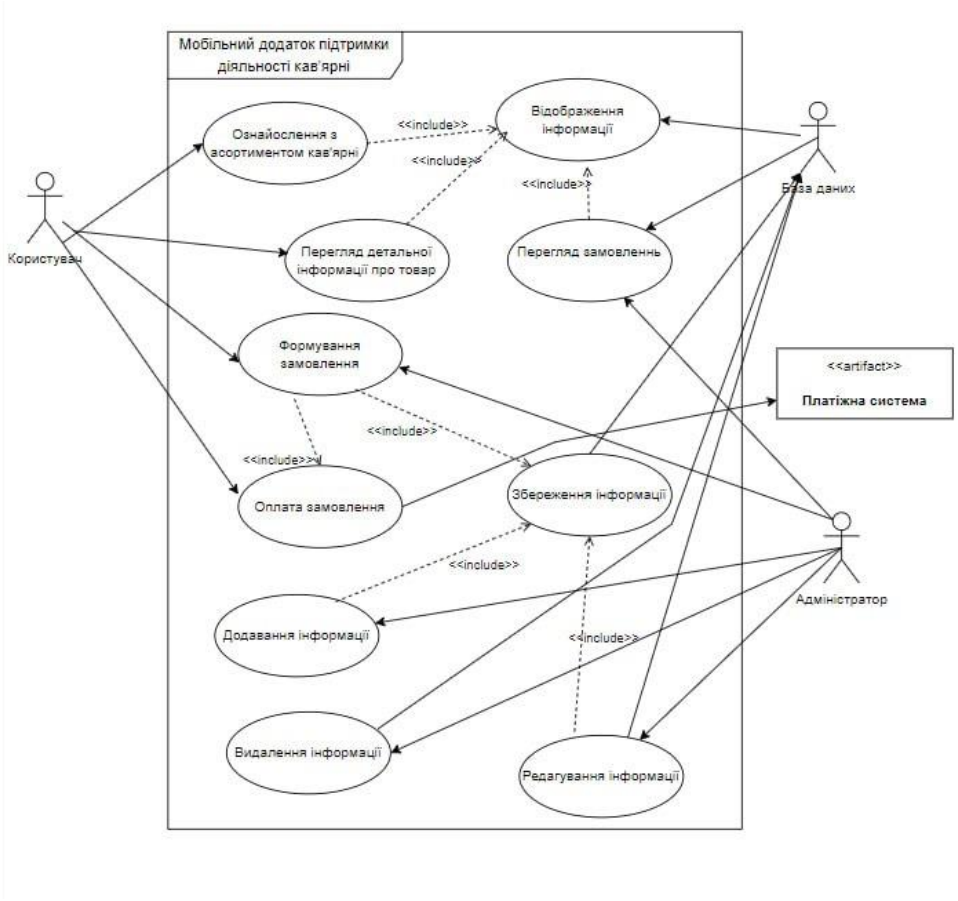

Рисунок 2.3 – Діаграма варіантів використання

Як видно з рисунку 2.3 для даного мобільного додатку було виділено наступні типи акторів:

- **Користувач**: може переглядати асортимент кав’ярні, переглядати детальну інформацію про товари, формувати замовлення, здійснювати оплату, переглядати свої замовлення, а також використовувати функції додавання, редагування та видалення інформації;
- **Адміністратор**: має розширені права доступу, включаючи перегляд і редагування інформації, управління даними про товари та послуги, а також контроль за виконанням замовлень;
- **База даних**: зберігає всю інформацію про товари, замовлення, користувачів та інші необхідні дані для функціонування додатку;
- **Платіжна система**: використовується для обробки платежів при оформленні замовлень.

Опис варіантів використання такий:

- **Ознайомлення з асортиментом кав’ярні**: користувачі можуть переглядати доступні товари, які пропонує кав’ярня;
- **Перегляд детальної інформації про товар**: користувачі можуть отримувати детальну інформацію про обраний товар, включаючи його характеристики, ціну та наявність;
- **Формування замовлення**: користувачі можуть створювати замовлення, додаючи товари до кошика та оформлюючи покупку;
- **Оплата замовлення**: користувачі можуть здійснювати оплату замовлень через інтегровану платіжну систему;
- **Перегляд замовлень**: користувачі можуть переглядати свої попередні та поточні замовлення, відстежувати статус виконання;
- **Збереження інформації**: мобільний додаток забезпечує збереження всіх даних про замовлення, товари та користувачів у базі даних;
- **Додавання інформації**: адміністратор може додавати нові товари до асортименту кав’ярні;
- **Редагування інформації**: адміністратор може змінювати інформацію про наявні товари;
- **Видалення інформації**: адміністратор може видаляти товари з асортименту.

Ця діаграма допомагає чітко визначити ролі користувачів та їх взаємодію з програмний продуктом, що є ключовим для успішного проєктування та подальшій реалізації мобільного додатку підтримки діяльності кав’ярні.

2.3 Проєктування моделі бази даних

Система управління базами даних (СУБД) PostgreSQL [26] – це одним із ключових інструментів даної розробки. Вона пропонує створення високошвидкісної, надійної та масштабованої бази даних (БД), яка забезпечує

ефективне збереження та обробку даних. СУБД PostgreSQL працює на основі технології SQL. Це дозволяє легко зберігати та опрацьовувати інформацію у вигляді структурованих таблиць. Вона також підтримує складні запити та транзакції, що забезпечує користувачам можливість виконувати певні операції з даними без втрати їх цілісності. PostgreSQL підтримує різноманітні типи даних, включаючи JSON[27], що робить її гнучкою для зберігання як традиційних реляційних даних, так і даних, які не підходять для традиційних схем.

Для ефективної роботи з інформацією даного мобільного додатку було вирішено використати базу даних із використанням СУБД PostgreSQL. Фізична модель цієї БД представлена на рисунку 2.4. Вона забезпечує зберігання інформації про зареєстрованих користувачів, їх замовлення, товари та інше. У представленій БД створено 9 таблиць, кожна з яких відповідає за окремий набір структурованої інформації.

Перелік таблиць у базі даних такий:

- admin – таблиця, що зберігає дані адміністраторів;
- refresh_token – таблиця для зберігання токенів оновлення;
- status – таблиця для зберігання статусів замовлень;
- order – таблиця для зберігання замовлень;
- order_item – таблиця для зберігання позицій замовлень;
- payment_method – таблиця для зберігання методів оплати;
- product – таблиця для зберігання інформації про продукти;
- image – таблиця для зберігання зображень продуктів;
- category – таблиця для зберігання категорій товарів;

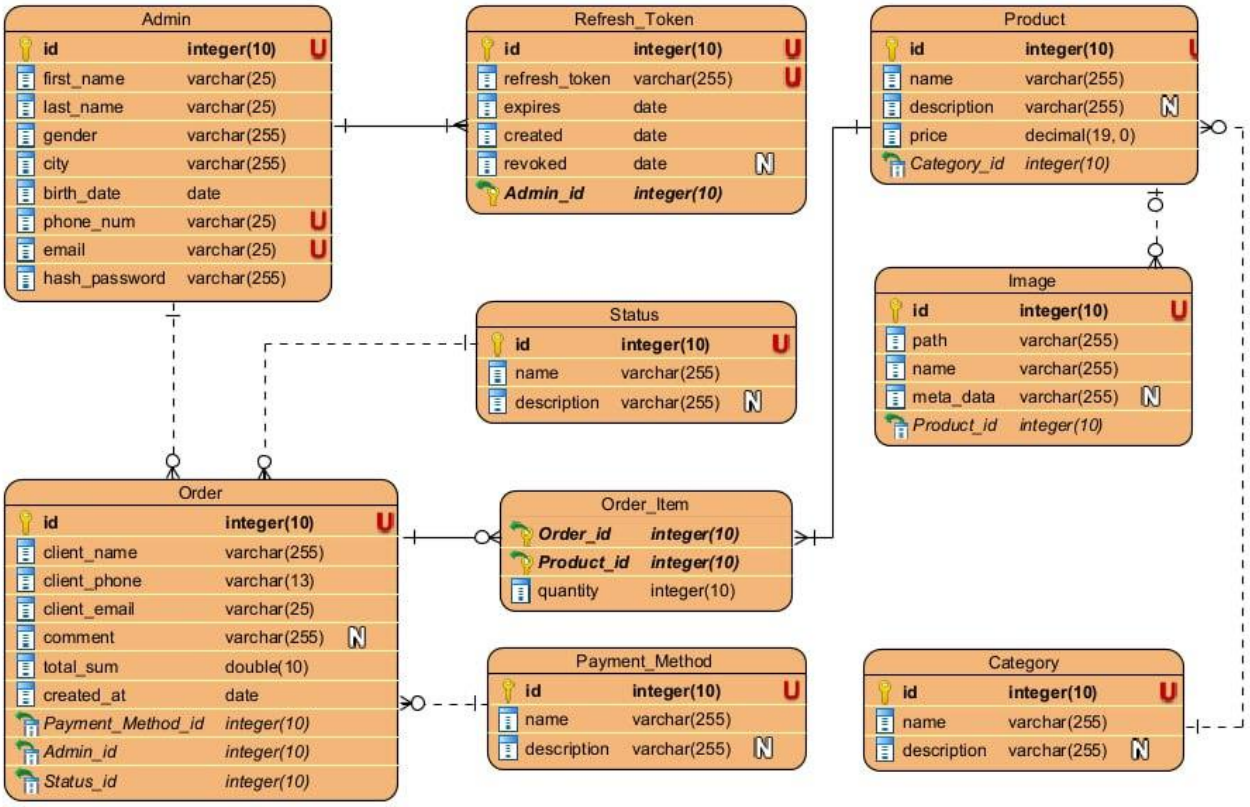

Рисунок 2.4 – Фізична модель бази даних

На рисунку 2.4 представлена фізична модель бази даних. Вона включає таблиці з відповідними полями, зв'язками між таблицями та основними ключами. Для зберігання даних було вирішено використовувати Amazon Web Services (AWS), що забезпечує високий рівень надійності, безпеки та масштабованості. Зокрема, для управління базою даних використовується Amazon RDS (Relational Database Service), який дозволяє автоматизувати процеси резервного копіювання, оновлення програмного забезпечення та масштабування.

Основні компоненти підключення:

- **Amazon RDS**: Використовується для управління та розгортання PostgreSQL бази даних. Amazon RDS забезпечує високу доступність та автоматичне масштабування;
- **Amazon EC2**: Використовується для розгортання серверної частини додатку, яка взаємодіє з базою даних;
- **PostgreSQL**: Основна СУБД для зберігання даних додатку.

3 ПРАКТИЧНА РЕАЛІЗАЦІЯ МОБІЛЬНОГО ДОДАТКУ

3.1 Архітектура мобільного додатку

Архітектура мобільного додатку – це структурний підхід до розробки програмного забезпечення (ПЗ). Він визначає спосіб організації компонентів, модулів та логіки програми для досягнення певних цілей ефективності, масштабованості, надійності та підтримуваності. Належна архітектура мобільного додатку дозволяє забезпечити зручний та стабільний функціонал для користувачів та спрощує створення та підтримку ПЗ для розробників.

Мінімально життєздатний продукт (MVP) – це версія продукту з достатньою кількістю функцій [28], щоб задовольнити перших клієнтів і забезпечити зворотний зв’язок для подальшого розвитку. При реалізації MVP з використанням архітектурних патернів Model-View-Controller (MVC) і Model-View-ViewModel (MVVM) [29] необхідно врахувати певні міркування, щоб забезпечити життєздатність і масштабованість продукту.

MVP зазвичай фокусується на основних функціональних можливостях, які відповідають основним потребам користувачів. Наприклад, у мобільному додатку підтримки діяльності кав’ярні MVP може включати такі функції, як перегляд кавових продуктів, додавання товарів до кошика та завершення покупки. Ці дії забезпечують основну функціональність, мінімізуючи час і ресурси розробки.

У ході створення даного мобільного додатку була обрана за основу саме ця архітектура. Її застосування дозволить розробити працюючий належним чином програмний продукт, який є неперевантаженим зайвими функціями. У свою чергу буде з використано архітектурний патерн Model-View-ViewModel (MVVM) (рис.3.1).

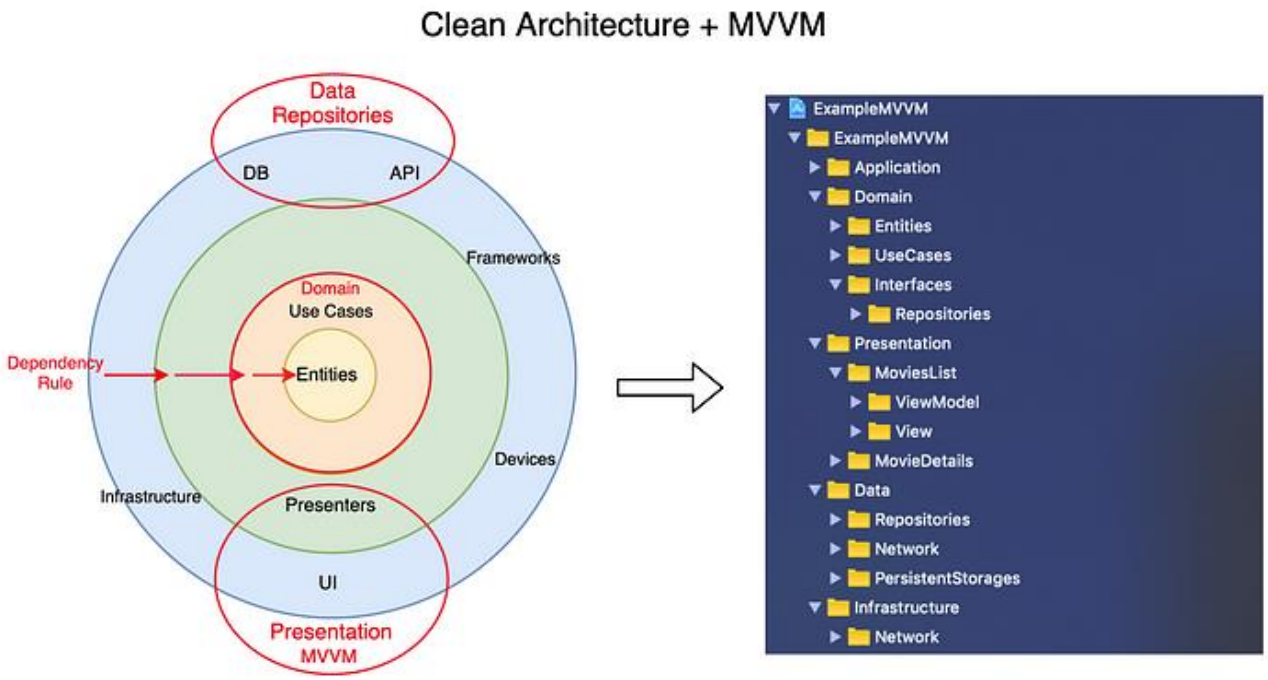

Рисунок 3.1 –Наявний приклад MVVM

Типова архітектура для мобільного додатку на React Native може базуватися на підході Clean Architecture [30], який розділяє додаток на окремі шари, щоб забезпечити високу розширюваність та тестованість коду. Вона включає такі шари:

1. **Presentation Layer (UI Layer):** цей шар містить всі компоненти інтерфейсу користувача, такі як екрани, компоненти та навігація. У React Native це можуть бути файли JSX [31], які мають компоненти, їх стилі та маршрутизацію за допомогою бібліотеки React Navigation.

2. **Domain Layer (Business Logic):** цей шар містить бізнес-логіку додатку, таку як операції з даними та логіка додатку. У React Native це можуть бути JS-файли, які мають класи або функції, що виконують різні завдання: обробку даних, валідацію введених даних та взаємодію з сервером через API.

3. **Data Layer (Data Access Layer):** цей шар містить всю логіку доступу до даних, таку як запити до локальної бази даних, маніпулювання даними та взаємодія з віддаленим сервером через API. У React Native це можуть бути файли, які мають класи або функції для взаємодії з локальною базою даних (наприклад, SQLite або Realm) та віддаленим сервером (через HTTP-запити або використання бібліотек, таких як axios).

4. **Infrastructure Layer:** цей шар містить всю інфраструктуру, необхідну для роботи додатку, таку як конфігураційні файли, залежності та зовнішні сервіси. У React Native це можуть бути файли налаштування, константи та сторонні бібліотеки.

Така архітектура дозволяє розділити логіку програми на окремі частини. Це спрощує здійснення розробки, тестування та підтримки додатку. Крім того, вона забезпечує масштабованість та гнучкість коду. Це дозволяє легко додавати новий функціонал та вносити зміни без значних зусиль.

Архітектура взаємодії:

- Клієнт (Мобільний додаток): Написаний на React Native, взаємодіє з сервером через GraphQL;
- Сервер (Node.js + Express.js): Обробляє запити від клієнта та взаємодіє з базою даних PostgreSQL;
- GraphQL: Мова запитів для API, яка дозволяє клієнтам запитувати саме ті дані, які їм потрібні, і нічого зайвого. Це зменшує обсяг переданих даних і покращує продуктивність додатку;
- Amazon EC2: Використовується для хостингу серверної частини додатку;
- База даних (PostgreSQL): Розміщена на Amazon RDS, зберігає усі необхідні дані для функціонування додатку.

Схема архітектури взаємодії продемонстрований на рисунку 3.2.

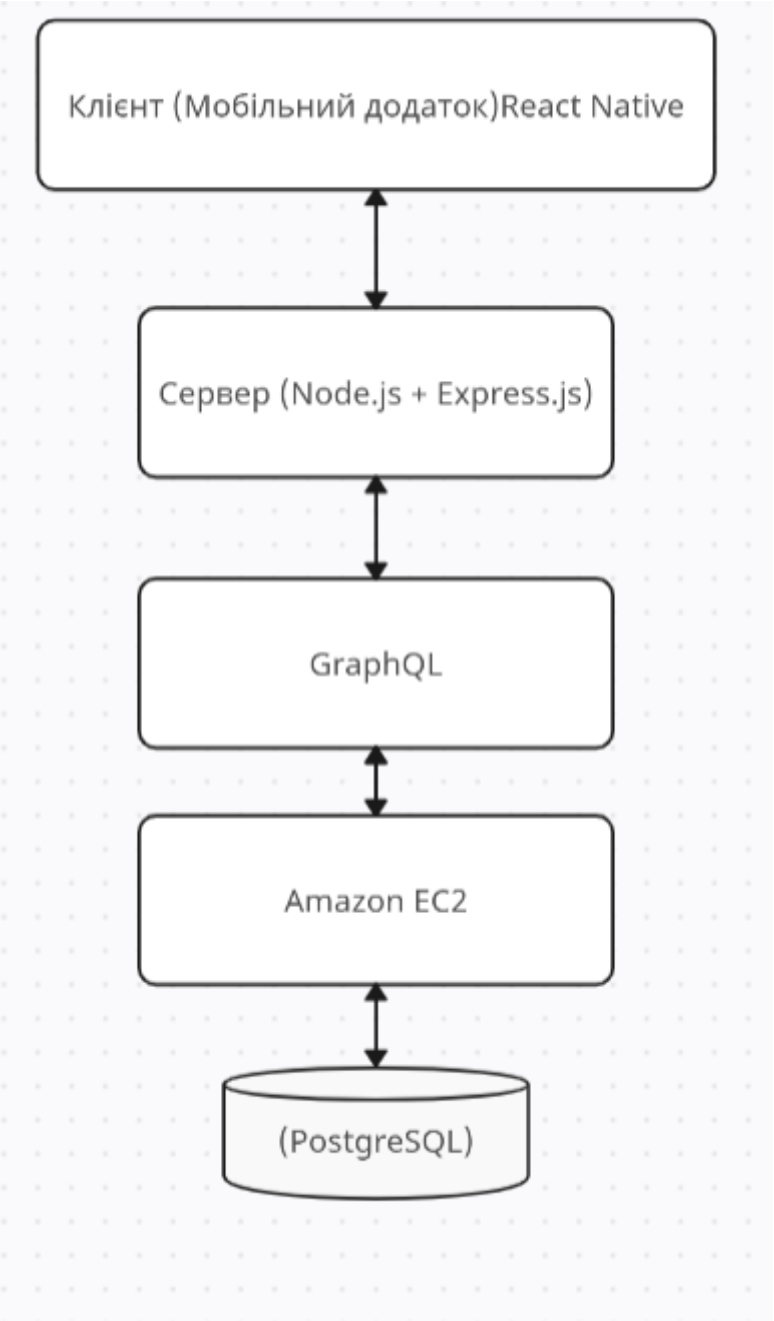

Рисунок 3.2 – Схема архітектури взаємодії

3.2 Розробка основних компонентів на React Native

Для розробки мобільного додатку було використано React Native, що дозволяє створювати кросплатформені мобільні додатки, які можуть працювати як на iOS, так і на Android. Основною перевагою цієї технології є можливість використання єдиного коду для обох платформ. Це значно скорочує час і витрати

на розробку. Нижче наведено опис деяких основних компонентів даного мобільного додатку та приклади їх реалізації.

HomeScreen – головний екран додатку, на якому відображається основна інформація про кав'ярню та меню доступних продуктів. Цей екран служить точкою входу для користувача, надаючи швидкий доступ до основних функцій додатку. Він включає навігаційне меню, список продуктів з їх зображеннями, назвами та короткими описами (рис. 3.3).

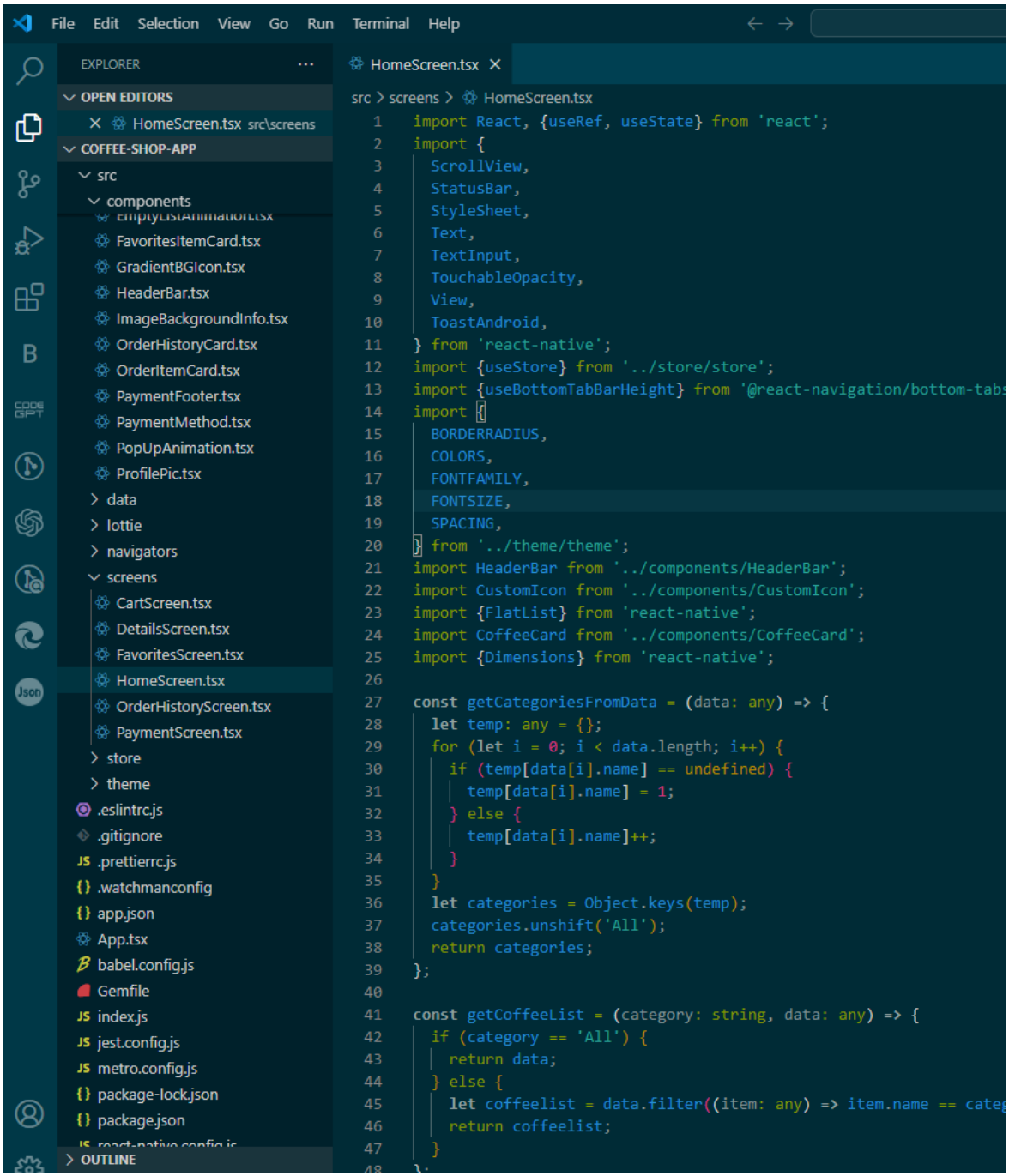

Рисунок 3.3 – Реалізація екрану «HomeScreen»

DetailsScreen – екран з детальною інформацією про обраний продукт. На цьому екрані користувач може переглянути більш детальний опис товару, включаючи його характеристики, відгуки інших клієнтів, можливість вибору розміру або типу продукту, а також додати його до кошика (рис. 3.4).

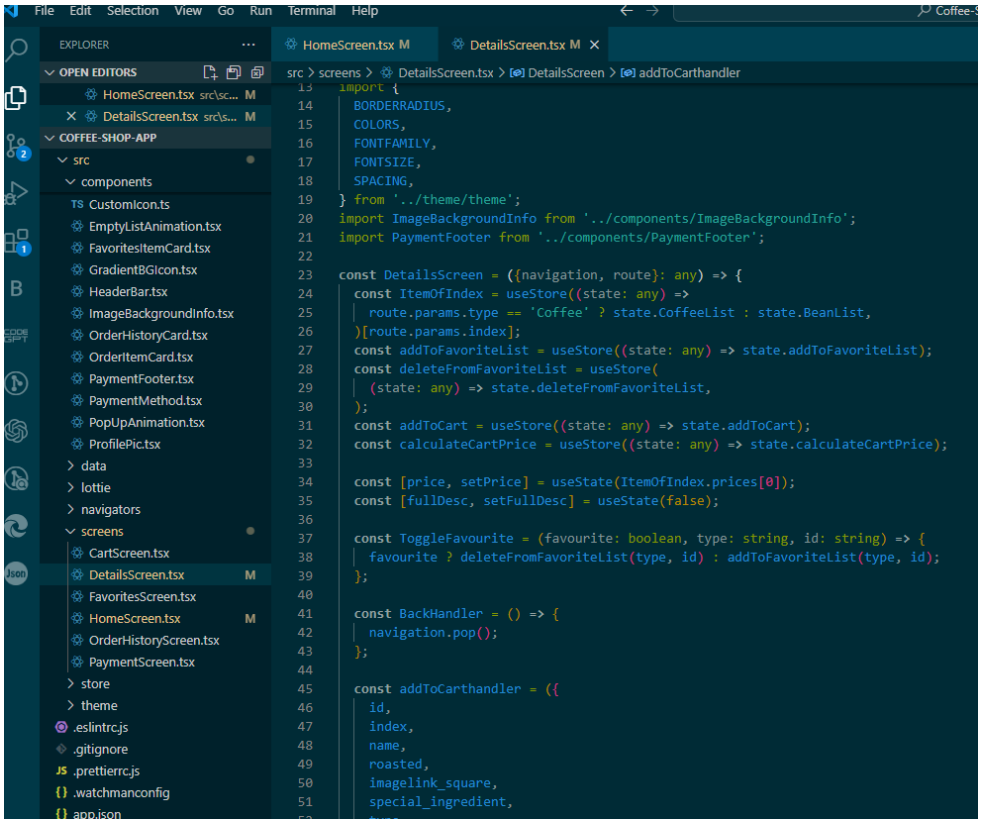

Рисунок 3.4 – Реалізація екрану «DetailsScreen»

CartScreen – екран кошика, де користувач може переглянути свої замовлення. Тут відображається список всіх доданих до кошика продуктів з можливістю змінення їх кількості або видалення. Також є можливість переходу до екрану оплати для завершення покупки (рис. 3.5).

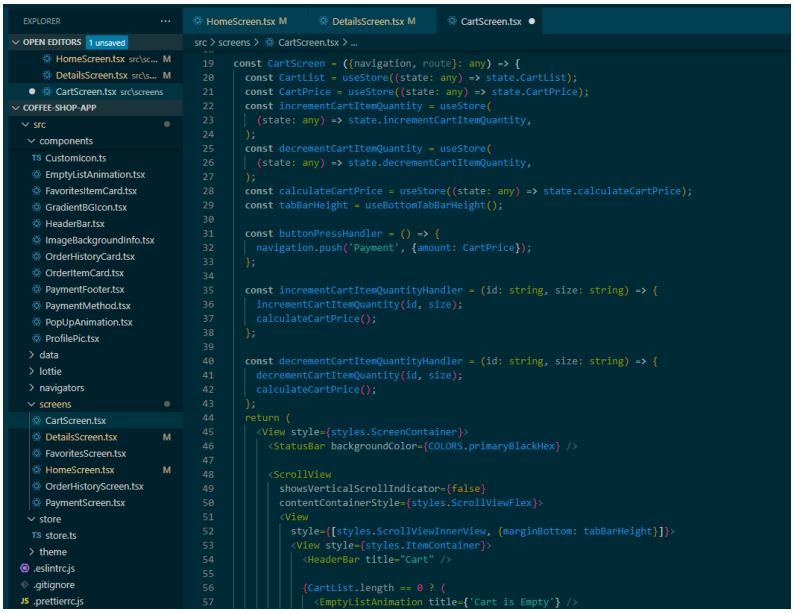

Рисунок 3.5 – Реалізація екрану «CartScreen»

PaymentScreen – екран оплати, де користувач вводить платіжні дані для завершення покупки. Він включає різні способи оплати, такі як кредитні карти, електронні гаманці та інші методи. Після успішної оплати користувач отримує підтвердження та інформацію про доставку свого замовлення на номер введеного мобільного телефону (рис. 3.6).

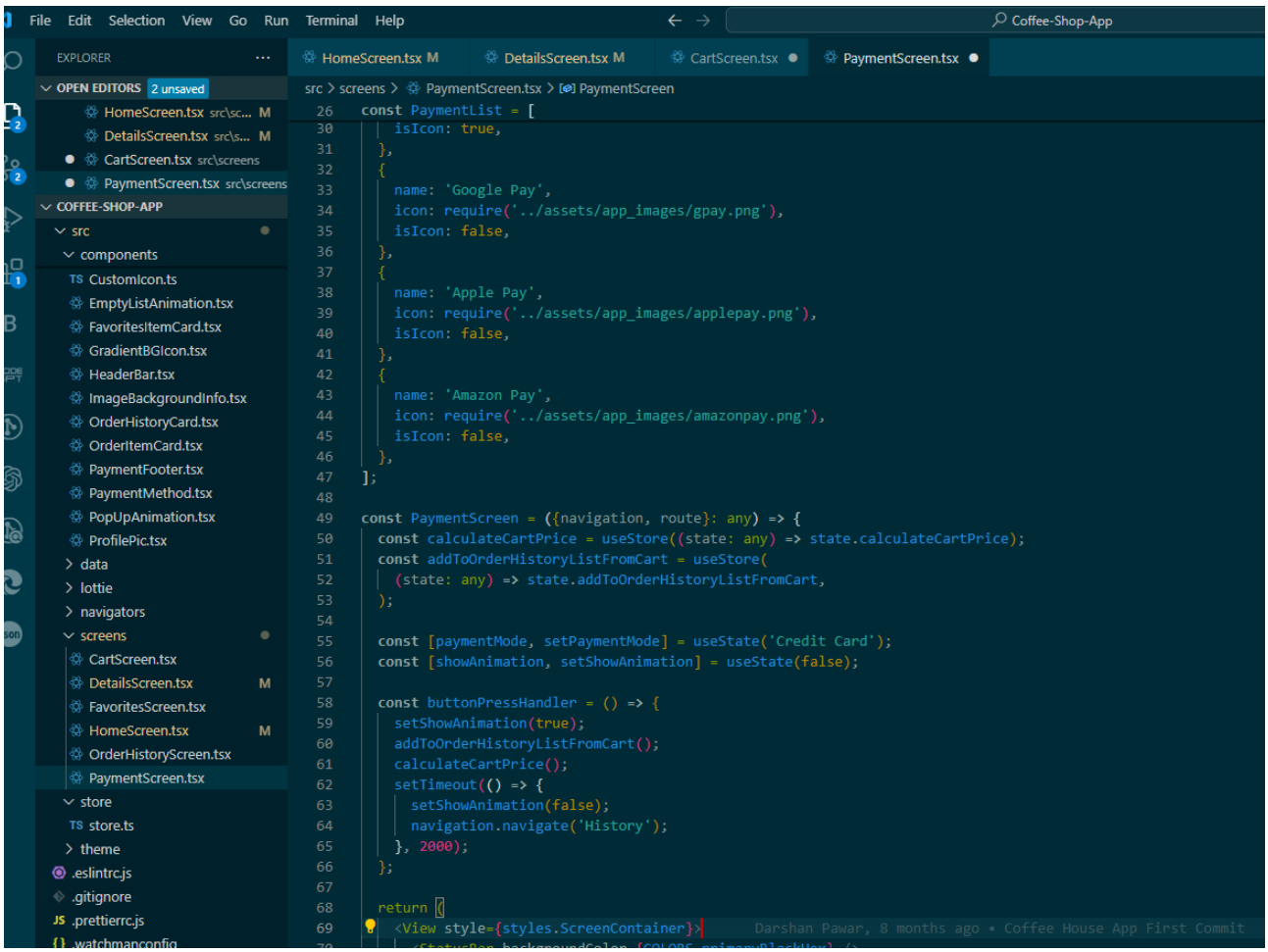

Рисунок 3.6 – Реалізація екрану «PaymentScreen»

Насправді таких компонентів, яких можна назвати основними, значна кількість. Тому на рисунку 3.7 зображено загальну кількість важливих компонентів даного мобільного додатку. Лістинг коду їх реалізації надано в Додатку В.

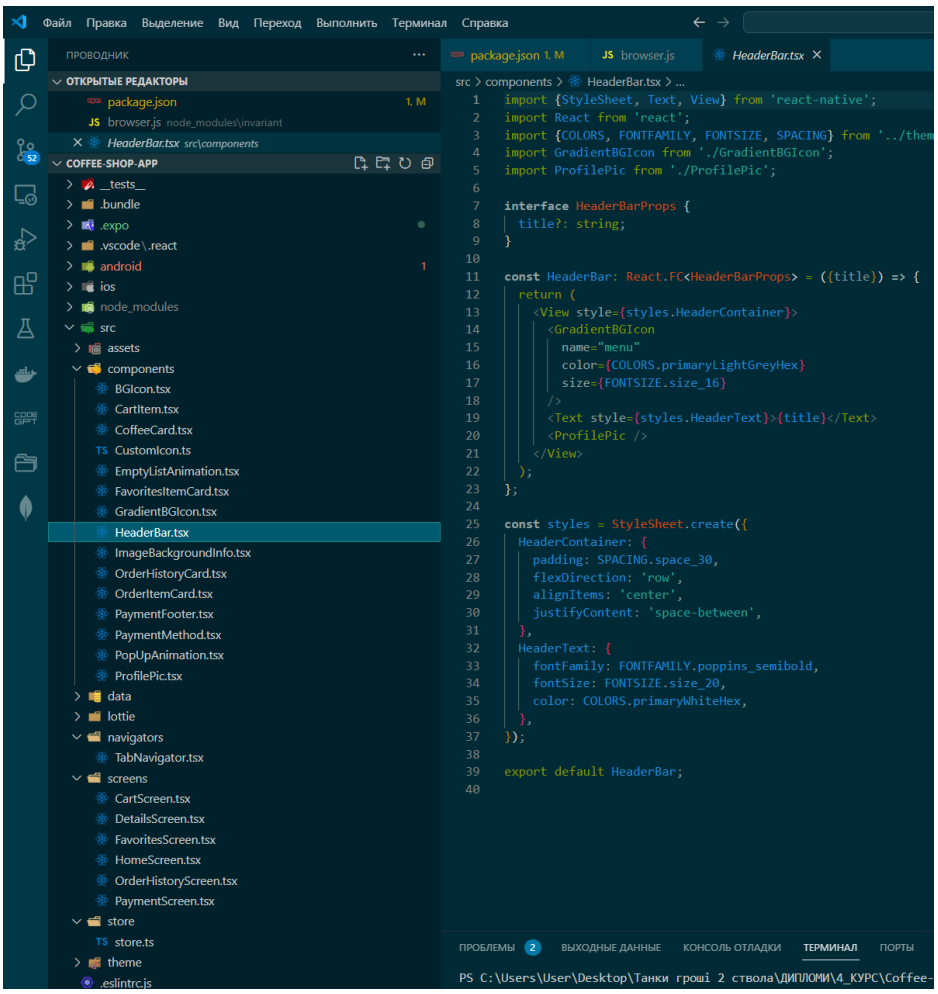

Рисунок 3.7 – Основні компоненти мобільного додатку

3.3. Демонстрація роботи мобільного додатку

Процес запуску даного мобільного додатку на React Native полягає у використанні різних інструментів та команд для побудови, збирання та запуску його на мобільному пристрої або емуляторі. Для майбутніх користувачів всі ці кроки будуть не потрібні. Їм просто треба завантажити представлений мобільний додаток. Після цього вони відразу можуть користуватися ним. Однак, у разі внесення змін розробником, вищезазначені дії є необхідними. Тому далі продемонстровано загальнний локальний запуск мобільного додатку. По-перше, перед початком роботи потрібно встановити всі необхідні залежності, включаючи Node.js, React Native CLI, Android SDK або Xcode під Android або

iOS відповідно. Після цього створюємо новий проєкт React Native за допомогою команди **npx react-native init ProjectName**. Після створення проєкту запускається емулятор для відображення додатку на віртуальному пристрої або підключити реальний мобільний пристрій до комп'ютера.

Потім запускається мобільний додаток за допомогою команди **npx react-native run-android** для Android або **npx react-native run-ios** для iOS. Після запуску додатку можемо переходити по його різним екранам та функціям для тестування його роботи. У випадку необхідності під час розробки може знадобитися налагодження та відлагодження помилок у додатку за допомогою вбудованих інструментів для розробки та налагодження, таких як Chrome DevTools для вебчастини додатка або Xcode/Android Studio для мобільних платформ.

Це загальний принцип запуску мобільного додатку на React Native, але конкретні кроки можуть відрізнятися в залежності від платформи розробки та налаштувань проєкту. Візуальний вигляд, та запуск проєкту показаний на рисунку 3.8.

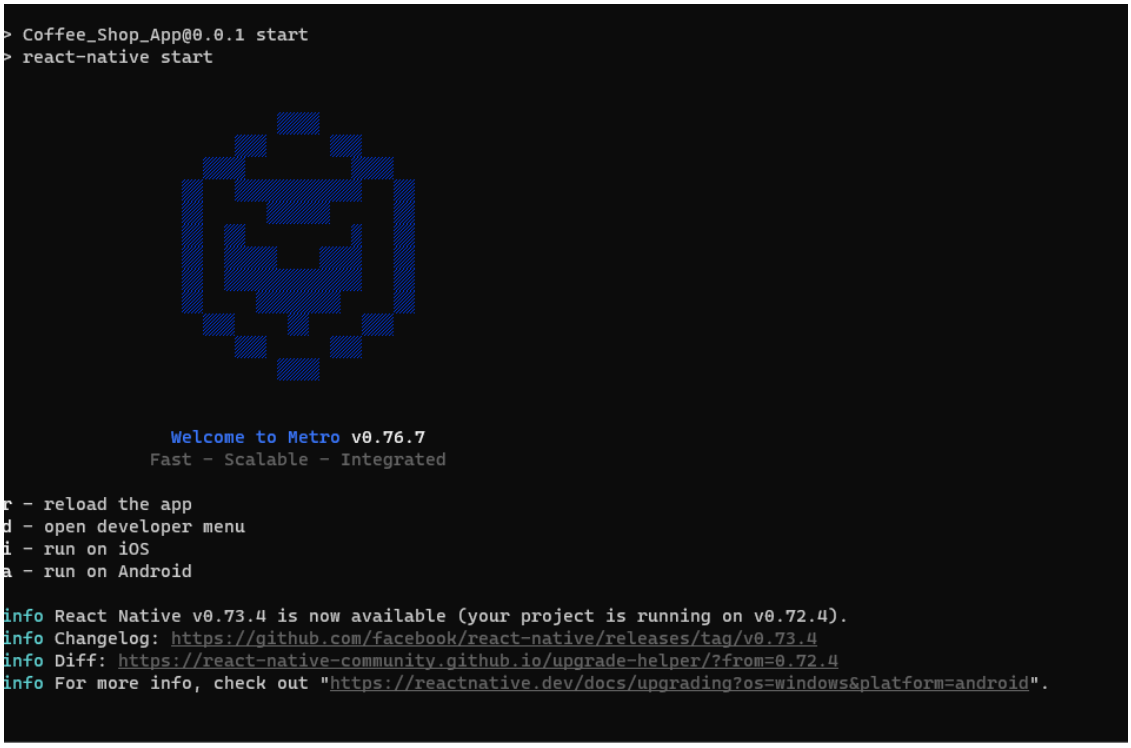

Рисунок 3.8 – Запуск мобільного додатку розробником

Зважаючи на роль кожної з сторінок у функціоналі мобільного додатку, варто детально розглянути кожну з них.

HomeScreen (Екран домашньої сторінки). Із цього екрану користувач зазвичай починає роботу з даним мобільним додатком (рис. 3.9). Основна мета цього екрану – представити асортимент товарів, які надає програмний продукт. Користувач прогортає список кавових сортів, кожен з яких представлений яскравим зображенням, назвою та коротким описом смаку або походження.

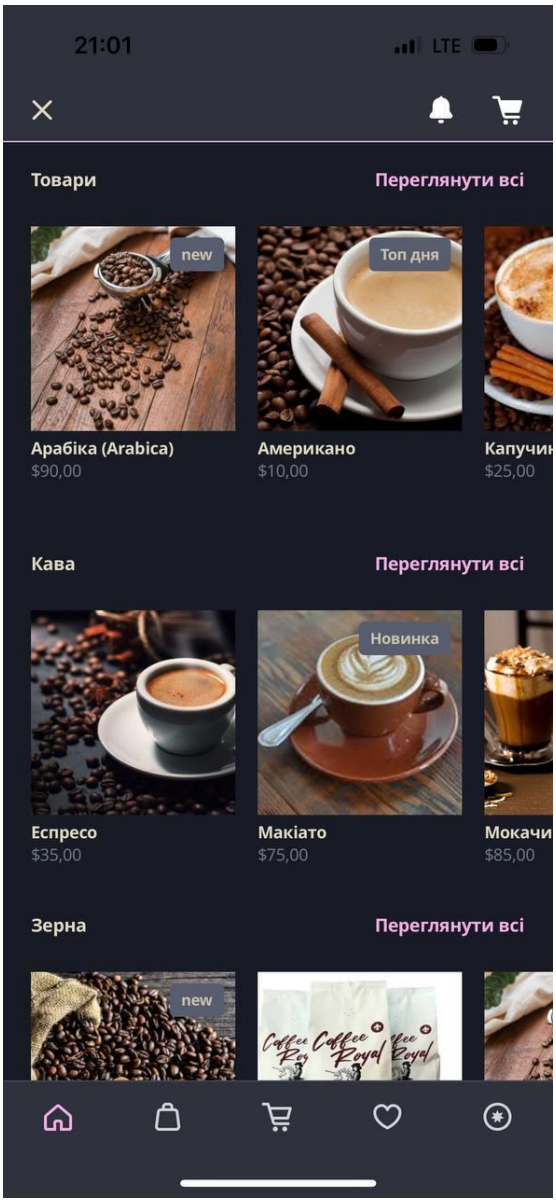

Рисунок 3.9 – HomeScreen

Details Screen (Екран деталей). Після вибору конкретного продукту на екрані домашньої сторінки, користувач переходить на екран деталей (рис. 3.10),

де може отримати більше інформації про обраний товар. На ньому розміщені більш докладні описи, відгуки клієнтів, можливість обрати розмір або зерна, а також інші характеристики. Останні допомагають користувачеві зробити свій вибір. У результаті обрані товари потрапляють у Кошик.

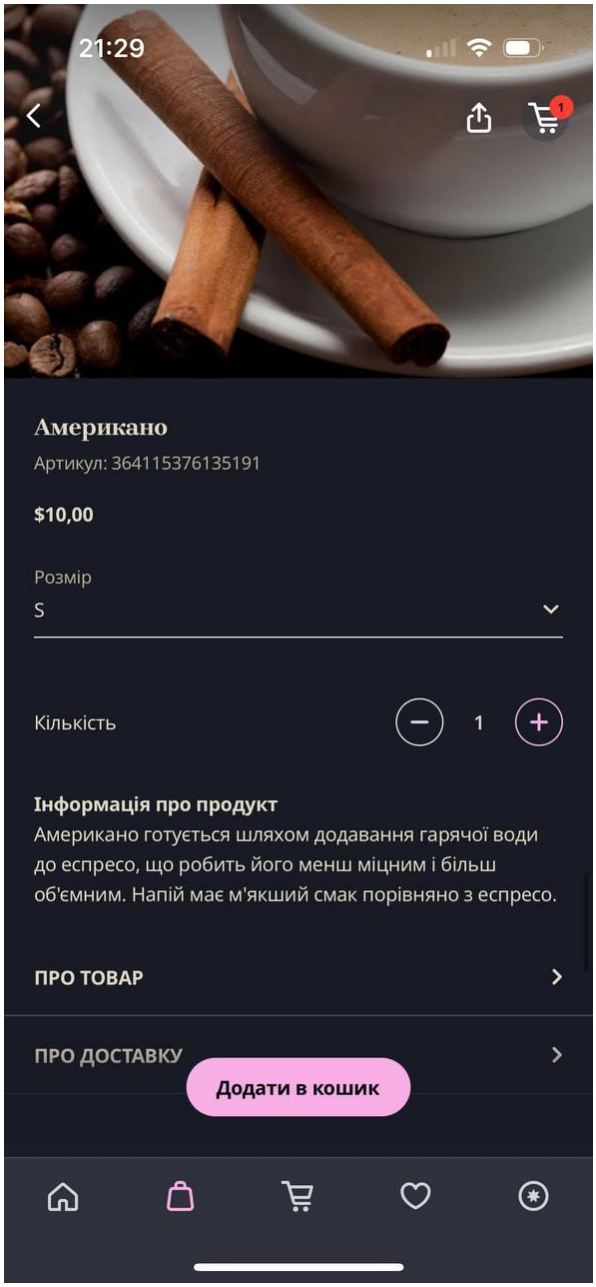

Рисунок 3.10 – Details Screen

Cart Screen (Екран кошика). У Кошику користувач може переглянути всі додані товари та вирішити, чи хоче він продовжити покупки та здійснити оформлення замовлення. Тут представлений загальний перегляд товарів у формі

списку або таблиці із можливістю змінити кількість товару, видалити їх або навіть зробити їх придбання в один клік (рис. 3.11).

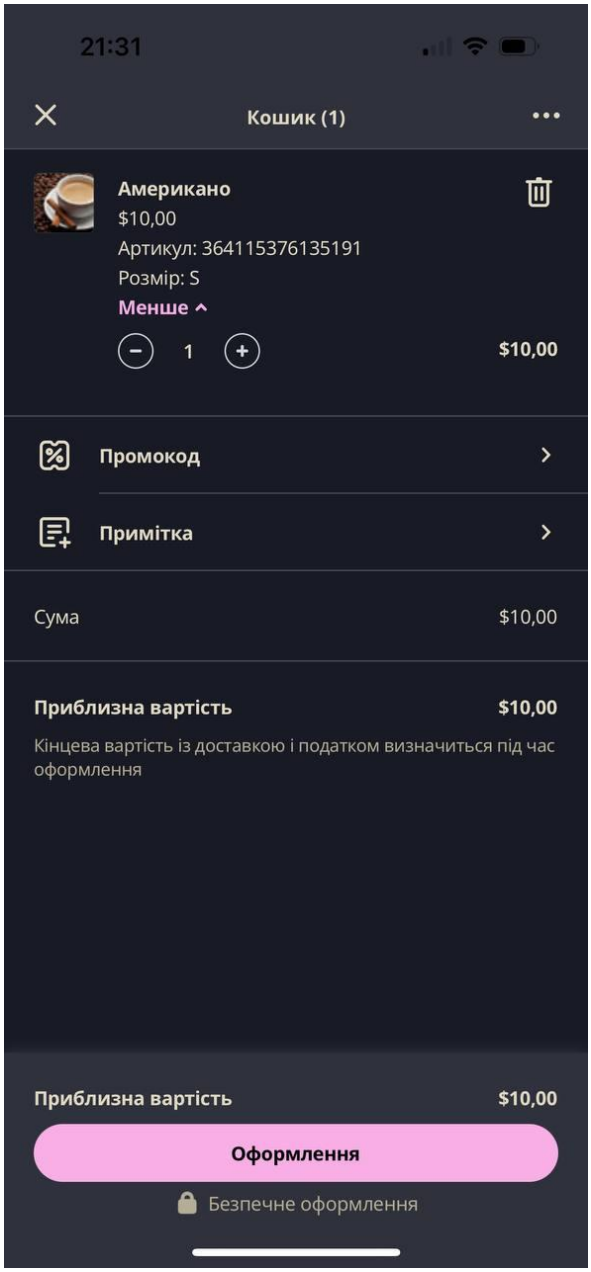

Рисунок 3.11 – Cart Screen

Payment Screen (Екран оплати). Фінальний крок перед завершенням покупки – екран оплати (рис. 3.12). Тут користувач вводить свої дані для оплати, такі як номер кредитної картки, дійсний термін, CVV-код тощо. Крім того, на ньому представлено різні способи оплати. Від кредитних карт до платіжних систем. Після успішної оплати користувач отримує підтвердження операції та

інформацію про доставку свого замовлення на введений раніше номер мобільного телефону.

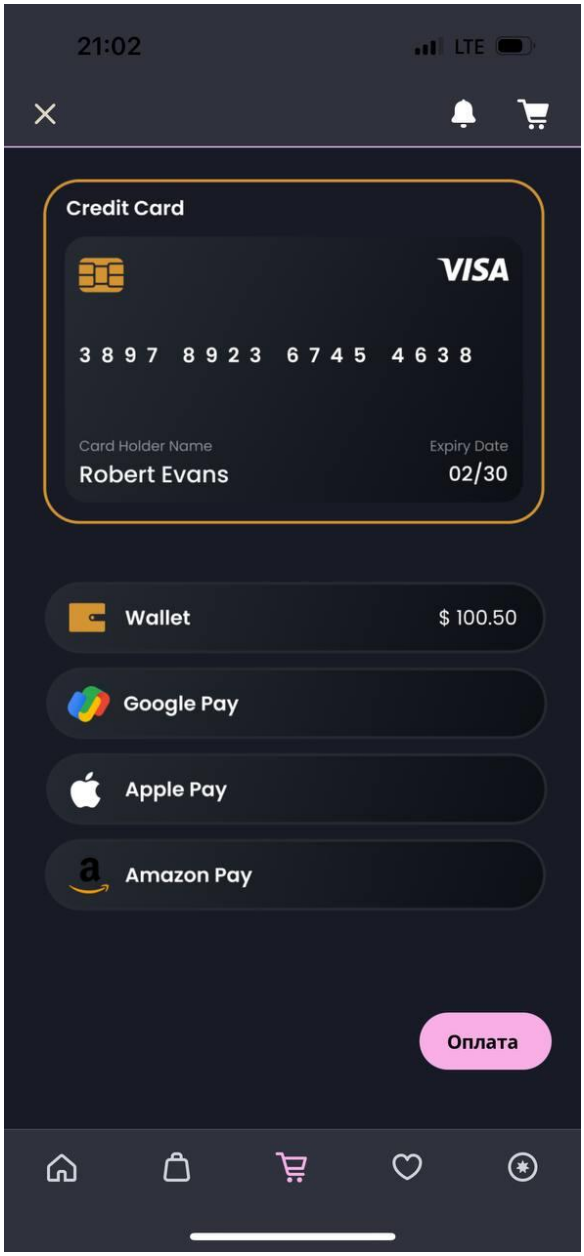

Рисунок 3.12 – Payment screen

Рисунок 3.13 надає загальний дизайн мобільного додатка. Він відображає загальний вигляд і стиль всього програмного продукту. Цей дизайн включає кольорову схему, шрифти, типографіку, стилі кнопок та інші елементи дизайну. Він відображає загальний вигляд і атмосферу додатка, створюючи однорідний та привабливий візуальний досвід для користувачів.

На рисунку 3.13 можна побачити загальну композицію додатка. Цей

дизайн включає основні елементи інтерфейсу. Це верхній бар із логотипом або меню, навігаційні кнопки, а також загальний розмітку сторінок.

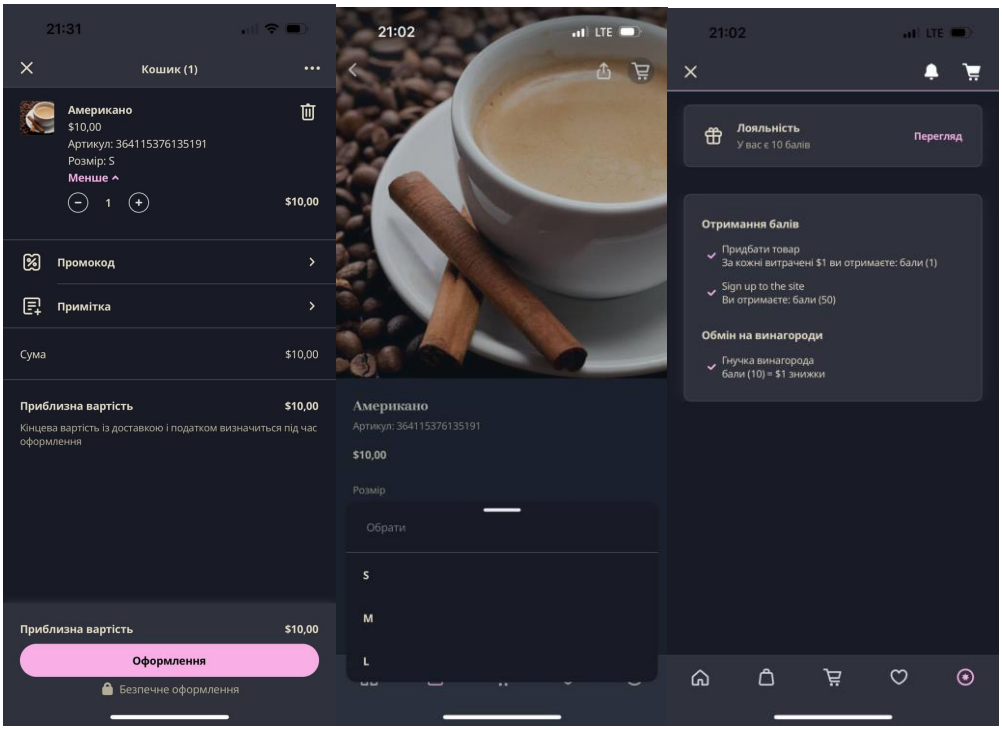

Рисунок 3.13 – Загальний дизайн мобільного додатка

Ці рисунки відображають результат розробки мобільного додатка підтримки діяльності кав’ярні, оскільки вони демонструють фінальний вид даного програмного продукту.

3.4 Тестування мобільного додатку

У даній роботі використовується тестування за допомогою Jest [37]. Це популярний тестовий фреймворк для JavaScript. Він забезпечує зручне написання та виконання тестів і підтримує автоматичне знаходження тестових файлів, мокінг об’єктів, знімкове тестування та асинхронне тестування.

Jest працює шляхом запуску тестових файлів та виконання визначених тестів. Кожен із них ізолюється. Це дозволяє уникнути взаємного впливу тестів один на одного. При завершенні виконання тестів, Jest генерує звіти з результатами, вказуючи кількість пройдених та невдалих тестів, час виконання та іншу корисну інформацію.

На рисунку 3.14 показано результат тестування одного з модулів проєкту. Воно було успішно пройдено. Тестування проводилося за допомогою команди jest, що запускає всі тестові файли в проєкті. Як видно з рисунку 3.14, тест App/index.test.js пройдено успішно (позначено як "PASS"). Усі наступні заплановані тести були виконані також без помилок. Це свідчить про правильну роботу розробленого мобільного додатку підтримки діяльності кав’ярні.

Тестування з Jest забезпечує високу надійність та стабільність коду, дозволяючи швидко виявляти та виправляти помилки на ранніх етапах розробки.

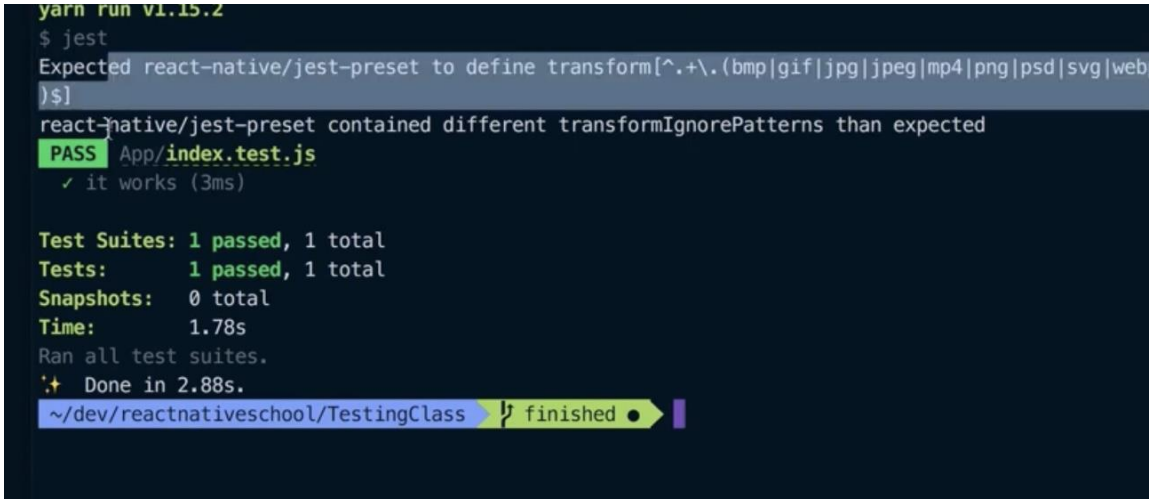

Рисунок 3.14 – Результат тестування

Код основних пройдених тестів для перевірки роботи даного мобільного додатку наведено в Додатку В.

ВИСНОВКИ

У ході виконання кваліфікаційної роботи бакалавра було досліджено предметну область шляхом проведення аналізу поточних тенденцій у розробці мобільних додатків підтримки діяльності сучасного бізнесу, зокрема кав’ярень, У результаті визначено актуальність даного проєкту. Також здійснено огляд відповідних аналогів. Виявлено їхні переваги та недоліки. Визначено ключові особливості та напрямки для їх урахування при подальшої розробці власного програмного продукту.

Визначено мету та задачі даного досліджено. Виділено вимоги до розробки власного мобільного додатку підтримки діяльності кав’ярні. Було визначено засоби реалізації даного проєкту.

Також було проведено структурно-функціональне моделювання даного програмного продукту та моделювання варіантів його використання.

У результаті здійснено практичну реалізацію мобільного додатку підтримки діяльності кав’ярні. Проведено успішне його тестування, жодних помилок у роботі виявлено не було.

Технічне завдання надано в Додатку А. Детальне планування робіт даного проєкту представлено в Додатку Б. Лістинг коду основних компонентів даного мобільного додатку, а також проведених тестів надано в Додатку В.

У майбутньому розроблений програмний продукт має значний потенціал для подальшого розвитку. Одним із напрямків є інтеграція додаткових платіжних систем для підвищення зручності користувачів. Також варто розглянути впровадження функцій штучного інтелекту для персоналізації рекомендацій. Іншим перспективним напрямком є розширення функціональності додатку для підтримки програм лояльності та бонусних систем. Це дозволить підвищити залученість клієнтів. Крім того, варто розглянути можливість інтеграції з соціальними мережами.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. "Mobile Application Development, Usability, and Security" by Sougata Mukherjea [Електронний ресурс] – Режим доступу: https://link.springer.com/book/10.1007/978-3-319-46259-1.

2. Mobile app usage - Statistics & Facts [Електронний ресурс] – Режим доступу: https://www.statista.com/topics/1002/mobile-app-usage/#topicOverview.

3. Big Nerd Ranch Guide (2021). "Android Programming: The Big Nerd Ranch Guide (4th Edition)" by Bill Phillips, Chris Stewart, and Kristin Marsicano [Електронний ресурс] – Режим доступу: https://www.amazon.com/Android-Programming-Ranch-Guide-Guides/dp/0135245125.

4. iOS Apprentice (8th Edition): Beginning iOS development with Swift by Joey deVilla, Ray Wenderlich Books (2021) [Електронний ресурс] – Режим доступу: https://www.amazon.com/iOS-Apprentice-Eighth-Beginning-Development/dp/1942878974.

5. "Web Application Development: Principles and Best Practices" by Sam Newman [Електронний ресурс] – Режим доступу: https://www.oreilly.com/library/view/web-application-development/9781492052187/.

6. "Hybrid App Development: Combining the Best of Both Worlds" by Laura Acklen [Електронний ресурс] – Режим доступу: https://www.techopedia.com/hybrid-app-development-combining-the-best-of-both-worlds/2/33822.

7. "HTML, CSS, and JavaScript - The Building Blocks of Web Development" by Mark Myers [Електронний ресурс] – Режим доступу: https://www.amazon.com/Web-Design-HTML-JavaScript-jQuery/dp/1118907442.

8. "Mastering Swift: Advanced Hands-On Programming" by Jon Hoffman [Електронний ресурс] – Режим доступу: https://www.amazon.com/Mastering-Swift-advanced-programming-development/dp/1789139864.

9. "Java and Kotlin for Android App Development" by Neil Smyth [Електронний ресурс] – Режим доступу: https://www.techotopia.com/index.php/Java_and_Kotlin_for_Android_App_Development.

10. "React Native: A Deep Dive into Mobile App Development" by John Capper [Електронний ресурс] – Режим доступу: https://www.apress.com/gp/book/9781484244531.

11. Starbucks [Електронний ресурс] – Режим доступу: https://www.linkedin.com/company/starbucks/Dunkin [Електронний ресурс] – Режим доступу: https://www.linkedin.com/company/dunkin’%E2%80%8B/.

12. Grubhub [Електронний ресурс] – Режим доступу: https://www.linkedin.com/company/grubhub-seamless/?trk=public_profile_experience-item_result-card_image-click&original_referer=https%3A%2F%2Fwww%2Egoogle%2Ecom%2F&originalSubdomain=ua.

13. dunkin [Електронний ресурс] – Режим доступу: https://www.linkedin.com/company/dunkin’%E2%80%8B/.

14. Navigating the App Store and Play Market: Strategies for Success" by Emily White [Електронний ресурс] – Режим доступу: https://www.digitaltrends.com/mobile/app-store-and-play-market-strategies-for-success/.

15. "Back-End Development: Architectures and Best Practices" by Alan Norton [Електронний ресурс] – Режим доступу: https://www.smashingmagazine.com/2021/07/backend-development-best-practices/.

16. "Front-End Web Development: The Big Nerd Ranch Guide" by Chris Aquino and Todd Gandee [Електронний ресурс] – Режим доступу: https://www.amazon.com/Front-End-Web-Development-Ranch-Guide/dp/0134433947.

17. "Learning React Native: Building Native Mobile Apps with JavaScript" by Bonnie Eisenman [Електронний ресурс] – Режим доступу: https://www.oreilly.com/library/view/learning-react-native/9781491929049/.

18. "Tailwind CSS: Rapid UI Development" by Adam Wathan [Електронний

ресурс] – Режим доступу: https://tailwindcss.com/docs.

19. "Node.js Design Patterns" by Mario Casciaro and Luciano Mammino [Електронний ресурс] – Режим доступу: https://www.packtpub.com/product/node-js-design-patterns-third-edition/9781839214110.

20. "Pro Express.js: Master Express.js for Your Web Development" by Azat Mardan [Електронний ресурс] – Режим доступу: https://www.apress.com/gp/book/9781484200385.

21. "Full Stack Mobile App Development with Ionic Framework, Firebase, and GraphQL" by Enoch O'Hara [Електронний ресурс] – Режим доступу: https://www.amazon.com/Full-Stack-Development-Framework-Firebase/dp/1839213989.

22. "Pro Git" by Scott Chacon and Ben Straub [Електронний ресурс] – Режим доступу: https://git-scm.com/book/en/v2.

23. Ярмак, В. В. "Методи моделювання процесів в інфраструктурних системах: IDEF0 та його варіації." Київ: Видавництво Наукова Думка, 2018.

24. Смирнов, О. О. "Теорія декомпозиції складних систем." Харків: Вид-во Харківського національного університету, 2019.

25. Павлов, М. І. "Діаграми варіантів використання в UML: Практичне керівництво." Львів: Видавництво Львівської політехніки, 2020.

26. Коваленко, А. П. "Бази даних PostgreSQL: Настанова для розробників." Дніпро: Дніпропетровський національний університет, 2021.

27. Гаврилюк, Д. О. "JSON: Формат обміну даними в інтернеті." Вінниця: Видавництво Вінницького національного технічного університету, 2017.

28. Петров, В. М. "Концепція мінімально життєздатного продукту: Теорія та практика." Одеса: Видавництво Одеського національного університету, 2020.

29. Сидоренко, І. В. "Архітектура MVVM в розробці програмного забезпечення." Чернівці: Видавництво Чернівецького національного університету, 2019.

30. Романенко, П. С. "Чиста архітектура: Підходи та принципи." Київ: Видавництво Київського політехнічного інституту, 2021.

31. Ткаченко, О. В. "JSX: Сучасний підхід до розробки інтерфейсів користувача." Запоріжжя: Видавництво Запорізького національного університету, 2018.

32. Іванов, Д. К. "DOM: Структура та маніпуляції з HTML-документами." Харків: Видавництво Харківського національного університету, 2019.

33. Лисенко, Н. М. "Розумні технології в програмній інженерії." Полтава: Видавництво Полтавського національного технічного університету, 2020.

34. Кравченко, В. О. "Структура розбиття робіт у проєктному менеджменті." Дніпро: Видавництво Дніпропетровського державного університету, 2019.

35. Сергєєв, М. П. "Організаційна структура розбиття: Теорія та практика." Суми: Видавництво Сумського державного університету, 2020.

36. Антонов, Ю. С. "Діаграма Ганта: Інструменти планування проєктів." Київ: Видавництво Київського національного університету, 2018.

37. "Jest: Delightful JavaScript Testing" by Simen Bekkhus [Електронний ресурс] – Режим доступу: https://jestjs.io/uk/docs/.

ДОДАТОК А

ТЕХНІЧНЕ ЗАВДАННЯ

на розробку проєкту:

«Мобільний додаток підтримки діяльності кав’ярні

Суми 2024

1. Призначення й мета мобільного додатку підтримки діяльності кав’ярні

1.1 Призначення мобільного додатку підтримки діяльності кав’ярні

Мобільний додаток призначений для підтримки діяльності кав’ярні. Він забезпечить ефективне управління замовленнями, взаємодією з клієнтами та удосконалить загальну організацію роботи відповідного закладу. Додаток має на меті створити зручний інтерфейс для користувачів, щоб спростити процес замовлення кави та інших напоїв, а також надати можливість взаємодії з програмою лояльності.

1.2 Мета створення мобільного додатку підтримки діяльності кав’ярні

Головна мета проєкту полягає у розробці кросплатформенного мобільного додатку підтримки діяльності кав’ярні. Його застосування спрямоване на автоматизацію важливих бізнес-процесів даного закладу, таких як обробка замовлень, управління взаємодією з клієнтами та використання програм лояльності. Це покращить якість обслуговування відвідувачів та забезпечить позитивний загальний користувальницький досвід.

1.3 Цільова аудиторія

Цільовою аудиторією мобільного додатку є клієнти кав’ярні, які цінують швидкість та зручність у замовленні кави та інших напоїв. Це включає в себе як регулярних відвідувачів, так і нових, які шукають зручний спосіб отримати свої улюблені напої. Особлива увага буде приділена молоді, яка активно використовує мобільні технології та цінує інновації.

2. Вимоги до проєкту розробки та впровадження мобільного додатку для кав’ярні

2.1 Вимоги до проєкту в цілому

2.1.1 Вимоги до структури й функціонування

Мобільний додаток підтримки діяльності кав’ярні повинен бути реалізований з використанням сучасних мобільних інструментів та забезпечувати визначений набір функціональних можливостей. Кінцевий продукт має містити якісне інформаційне наповнення та графічні матеріали, адаптовані під мобільні пристрої. Додаток повинен включати функції оформлення замовлення, перегляду меню, доступу до програми лояльності та можливість використання зворотного зв’язку з клієнтами.

2.1.2 Вимоги до персоналу

Персонал кав’ярні повинен мати здатність легко взаємодіяти з мобільним додатком без потреби у спеціалізованих технічних навичках. Основними вимогами є знання основ користування мобільними пристроями та здатність застосовувати базові функції додатку для обслуговування клієнтів і управління замовленнями.

2.1.3 Вимоги до збереження інформації

Усі дані, що вносяться та обробляються в мобільному додатку мають зберігатися в безпечній та надійній базі даних. Використання хмарних технологій для зберігання інформації забезпечить гнучкість та масштабованість. Важливим аспектом є також забезпечення конфіденційності та захисту особистих даних користувачів відповідно до законодавчих вимог.

2.1.4 Вимоги до розмежування доступу

Мобільний додаток повинен мати декілька рівнів доступу: адміністратор, працівник кав’ярні та клієнт. Адміністратор має повний доступ до усіх функцій додатку, включаючи управління контентом, меню та замовленнями. Працівники кав’ярні мають доступ до системи замовлень та обслуговування клієнтів. Клієнти можуть переглядати меню, робити замовлення та використовувати програму лояльності. Усі рівні доступу мають бути чітко визначені та захищені відповідними механізмами авторизації та аутентифікації.

2.2 Структура мобільного додатку

2.2.1 Загальна інформація про структуру мобільного додатку

Мобільний додаток підтримки діяльності кав’ярні повинен включати в себе всі необхідні функціональні сторінки доступні для загального користувача, а також адміністративну панель для використання відповідним персоналом кав’ярні.

Перелік сторінок мобільного додатку наступний:

- **«Головна» сторінка**: включає навігаційне меню, карусель з акційними пропозиціями, форму зв’язку з адміністрацією, кнопки переходу до особистого кабінету та оформлення замовлення;
- **«Меню» сторінка**: представляє асортимент кавових напоїв і десертів із можливістю переходу до деталей кожної позиції;
- **«Кастомізація напою»**: дозволяє користувачу персоналізувати своє замовлення, вибираючи інгредієнти та розмір;
- **«Забронювати столик»**: включає інтерактивну схему розміщення столів для бронювання;
- **«Доставка і оплата»**: інформація про умови доставки та способи оплати;
- **«Контакти»**: контактна інформація та мапа розташування кав’ярні;

– **«Особистий кабінет»**: історія замовлень користувача, улюблені напої та налаштування профілю.

2.2.2 Навігаційне меню

Навігаційне меню має бути простим і інтуїтивно зрозумілим, забезпечуючи легкий доступ до всіх основних розділів мобільного додатку. Меню повинно бути постійно доступним у верхній частині екрану для зручності користувачів.

2.2.3 Управління контентом

Управління контентом мобільного додатку має відбуватися через адміністративну панель, доступну відповідному персоналу кав’ярні. Усі дані, включаючи меню, акційні пропозиції та інформацію про продукти, мають зберігатися в надійній базі даних.

2.2.4 Дизайн мобільного додатку

Дизайн мобільного додатку повинен бути сучасним та стильним, відображаючи бренд та атмосферу кав’ярні. Використання фірмових кольорів: hsl(233, 79%, 53%), hsl(129, 7%, 20%), hsl(186, 87%, 27%) шрифта **Raleway** та графічних елементів має створювати приємне враження та забезпечувати зручність користування. Макет стилізованої сторінки даного мобільного додатку зображено на рисунку А.1.

Рисунок А.1 – Макет стилізованої головної сторінки

2.2.4 Система навігації (карта web-додатку)

Карта даного мобільного додатку зображена на рисунку А.2.

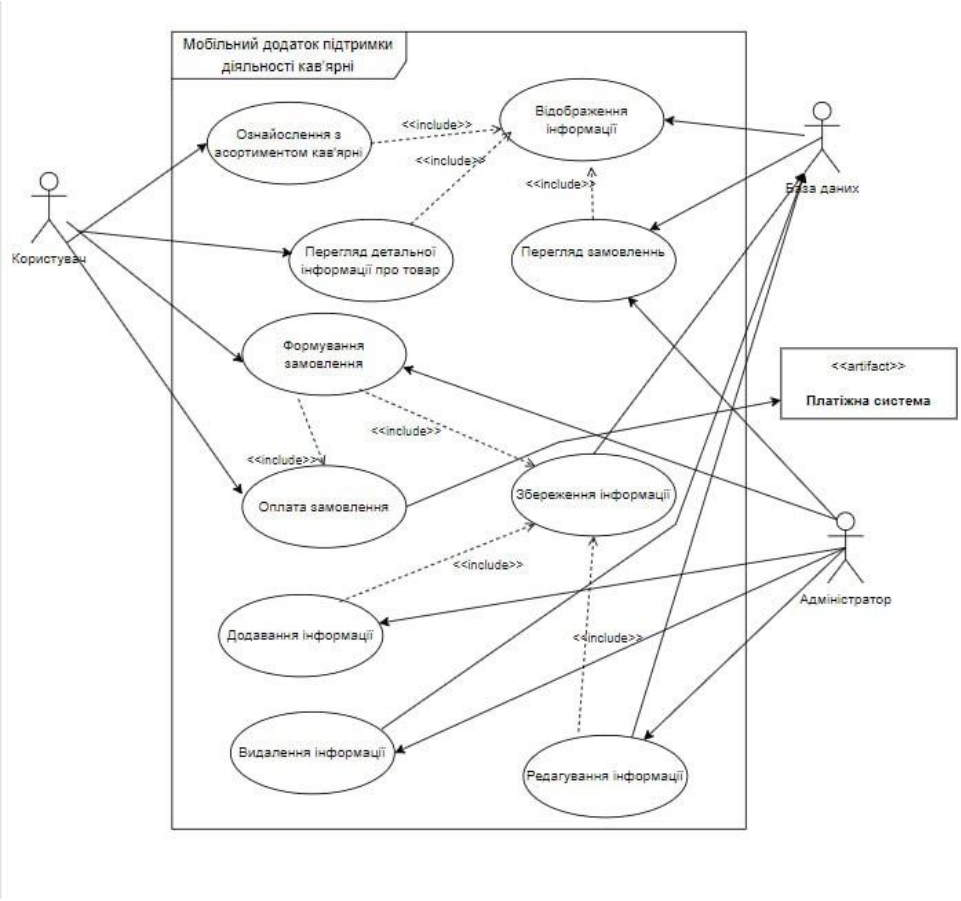

Рисунок А.2 – Система навігації

2.3 Вимоги до видів забезпечення

2.3.1 Вимоги до лінгвістичного забезпечення

Усі тексти в мобільному додатку мають бути доступні як мінімум на двох мовах: українською та англійською. Це забезпечить зручність користування для широкого кола клієнтів, включаючи місцевих жителів та іноземних відвідувачів. Переклад текстів повинен бути якісним і відповідати культурним та контекстуальним особливостям обох мов.

2.3.2 Вимоги до програмного забезпечення

Мобільний додаток має бути сумісним із широким спектром сучасних мобільних операційних систем, включаючи iOS та Android. Також важливо підтримувати останні версії основних мобільних браузерів, таких як Safari, Google Chrome, Mozilla Firefox та ін. для забезпечення стабільної роботи та

зручності користувачів.

2.4 Вимоги до функціонування системи

2.4.1 Потреби користувача

На базі потреб замовника та відгуків клієнтів було визначено основні потреби користувачів розроблюваного мобільного додатку (табл. А.1).

Таблиця А.1 – Потреби користувача

ID	**Потреби користувача**	**Функціонал**
UN-01	Перегляд меню	Клієнти та відвідувачі мають можливість переглядати асортимент продукції кав’ярні.
UN-02	Онлайн замовлення	Можливість здійснення онлайн-замовлення напоїв і десертів.
UN-03	Історія замовлень	Клієнти можуть переглядати історію своїх попередніх замовлень.
UN-04	Інформація про продукцію	Детальний опис напоїв і десертів.
UN-05	Акції та оголошення	Перегляд актуальних акційних пропозицій та оголошень.
UN-06	Інформація про кав’ярню	Відомості про кав’ярню, її місцезнаходження та контактні дані.
UN-07:	Бронювання столика	Онлайн-бронювання столика в кав’ярні.
UN-08:	Персоналізація замовлення	Можливість створення індивідуального замовлення за власними уподобаннями.
UN-09:.	Управління даними	Функціонал для адміністратора, що дозволяє керувати даними в мобільному додатку.
UN-10: персоналу.	Автоматизоване планування графіку роботи	Функція для адміністратора для автоматизованого планування роботи персоналу.

2.4.2 Функціональні вимоги

На основі аналізу потреб користувачів та персоналу кав'ярні було визначено наступні функціональні вимоги до розробки даного мобільного додатку:

- реалізація підтримки роботи на різноманітних девайсах для зручності всіх користувачів;
- забезпечення гнучкості у налаштуваннях, що дозволить адаптувати даний мобільний додаток під конкретні потреби кав'ярні, включаючи кастомізацію меню;
- реалізація інтеграції з існуючими pos-системами для удосконалення процесу управління замовленнями та звітністю;
- забезпечення підтримки програм лояльності та персоналізованих рекомендацій для заохочення клієнтів;
- реалізація можливості зворотного зв'язку для підтримки користувачів, щоб швидко реагувати на їх запити та вирішувати проблеми.

3. Склад і зміст робіт зі створення мобільного додатку підтримки діяльності кав'ярні

Детальний опис етапів створення даного мобільного додатку наведено в таблиці А.2.

Таблиця А.2 – Етапи створення мобільного додатку

№	Склад і зміст робіт	Строк розробки
1	Аналіз предметної області та визначення вимог	2 дні
2	Прототипування інтерфейсу користувача	5 дні
3	Розробка дизайну UI/UX	5 дні
4	Первинне програмування основних функцій мобільного додатку	5 дні
5	Розробка і інтеграція бекенду	10 днів

6	Тестування інтерфейсу та користувацького досвіду	3 дні

Продовження табл. А.2

№	**Склад і зміст робіт**	**Строк розробки**
7	Розширене тестування функціоналу (включаючи навантаження)	5 дні
8	Інтеграція з системами оплати та POS-системами	5 дні
9	Підготовка до запуску та розміщення у магазинах додатків	5 дні
10	Створення документації та маркетингових матеріалів	5 дні
	Загальна тривалість робіт	**50 днів**

4 Вимоги до складу й змісту робіт із введення мобільного додатку в експлуатацію

Вимоги до складу й змісту робіт із введення мобільного додатку в експлуатацію включають підготовку документації, проведення тестування власної розробки на різних пристроях і операційних системах, а також забезпечення належного рівня безпеки та налаштування зручного інтерфейсу для користувачів.

ДОДАТОК Б

Планування робіт

У сучасному світі швидкого розвитку цифрових технологій, мобільні додатки стали не лише зручним інструментом для користувачів, а й потужним каналом залучення нових клієнтів для бізнесу. У контексті гостинності, застосування спеціально створених мобільних рішень дозволяють кав'ярням значно підвищити рівень обслуговування, удосконалити роботу з відвідувачами та збільшити продажі. Розробка мобільного додатку підтримки діяльності кав'ярні є відповіддю на виклики сучасного ринку, де висока конкуренція вимагає інноваційного підходу до кожного аспекту бізнесу.

Актуальність даного проєкту полягає в необхідності адаптації до змінних уподобань споживачів, які воліють швидкість, зручність та персоналізацію. Мобільний додаток вирішено розробляти з метою задовольнити ці потреби, пропонуючи користувачам інтуїтивний інтерфейс для замовлення їхнього улюбленого кавового напою з будь-якого місця та в будь-який час.

Деталізація мети проєкту з використанням методу SMART [33]. Формулювання мети даного проєкту за SMART методом наступне: «Розробити мобільний додаток підтримки діяльності кав'ярні з реалізованими функціями оформлення замовлення клієнтами, перегляду меню, накопичування бонусів і бронювання столиків, спрямованого на підвищення лояльності клієнтів та забезпечення зручності їх обслуговування до 1 червня 2024 року».

Результати цього представлення мети проєкут за методом SMART надано в таблиці Б.1.

Таблиця Б.1 – SMART-мета розробки даного мобільного додатку

Specific	Розробити мобільний додаток підтримки діяльності кав'ярні, що надає можливість онлайн-замовлення, перегляду меню, управління бонусами та отримання персоналізованих пропозицій.
Measurable	Розроблений мобільний додаток.
Achievable	Є затверджене технічне завдання.
Relevant	Підвищити лояльність клієнтів та забезпечення зручності їх обслуговування.
Time-framed	Завершити проєкт до 1 червня 2024 року.

Планування змісту робіт. WBS (Work Breakdown Structure – ієрархічна структура робіт) [34]. WBS представляє собою графічний вигляд елементів проєкту, які групуються в ієрархічну структуру, що призводить до створення кінцевого продукту – мобільного додатку. Структура декомпозиції робіт спрямована на детальне виконання робіт по частинах і є ключовою у проєктуванні та організації командної роботи. Основні елементи декомпозиції включають розробку інтерфейсу, функціональності, бази даних та інтеграцію з існуючими системами кав'ярні. WBS забезпечує основу для оцінки термінів, витрат і графіків роботи.

На найвищому рівні WBS розміщується кінцевий продукт – мобільний додаток підтримки діяльності кав'ярні. Далі, процес розбивається на основні компоненти, такі, як дизайн, розробка, тестування та впровадження. Кожен з цих компонентів далі декомпонується на більш дрібні завдання, які стають елементарними роботами. Кожне елементарне завдання має чітко визначений результат та обсяг праці, що дозволяє призначити йому відповідальну особу та задати терміни виконання. WBS-структура робіт даного проєкту зображена на рисунку Б.1.

Організаційна структура виконавців. Створення OBS (Organizational Breakdown Structure) є наступним кроком після декомпозиції процесів [35]. OBS

визначає співробітників, відповідальних за виконання кожного елементарного завдання, визначеного у WBS. Кожне завдання може розглядатися як окремий міні-проєкт у рамках загального проєкту розробки мобільного додатку. OBS-структура робіт проєкту зображена на рисунку Б.2.

Таким чином, планування робіт та організаційна структура проєкту розробки мобільного додатку підтримки діяльності кав’ярні включає чітке визначення завдань, розподіл відповідальності та встановлення термінів, і виконавців (табл. Б.2), що забезпечує ефективне управління проєктом і його успішну реалізацію.

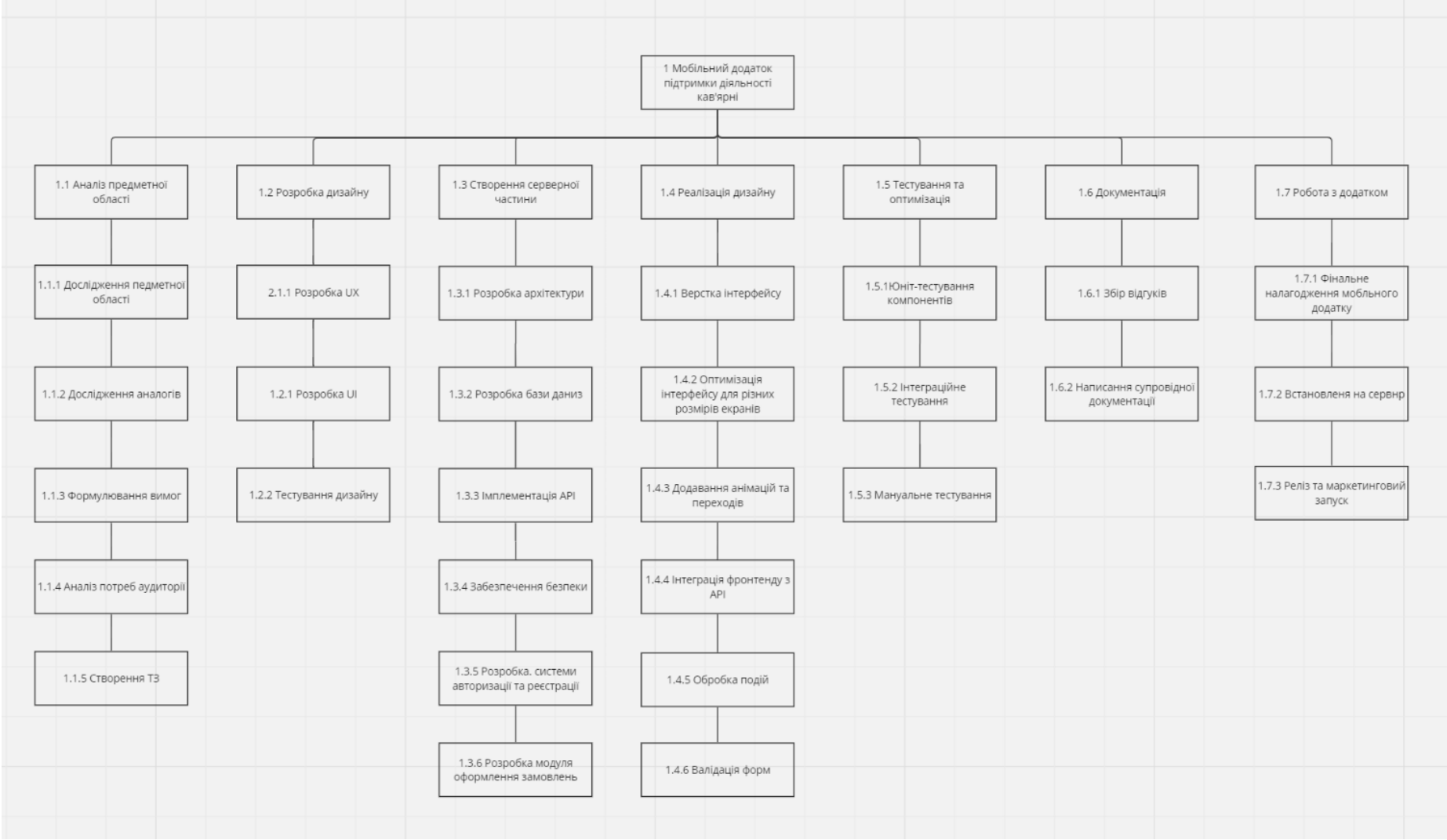

Рисунок Б.1 – WBS-структура робіт проєкту

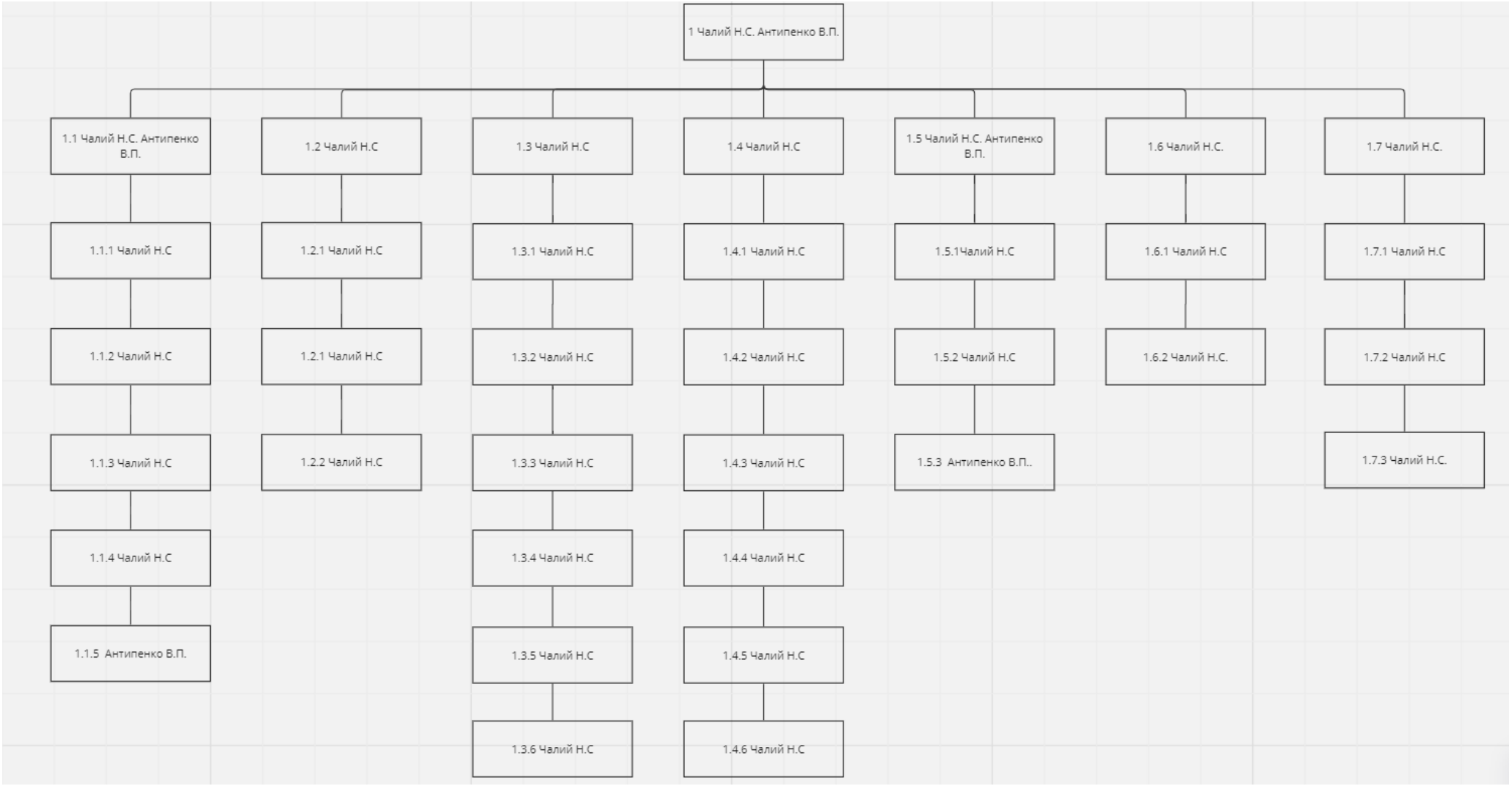

Рисунок Б.2 – OBS-структура робіт проєкту

Таблиця Б.2 – Виконавці проєкту

Роль	ПІБ	Проєктна роль
Розробник	Чалий Н.	Виконує розробку проєкту
Проєктувальник	Чалий Н.	Виконує проєктування бази даних та розробляє структуру мобільного додатку
Тестувальник	Чалий Н. Антипенко В.	Відповідає за тестування функціоналу
Керівник проєкту	Антипенко В.	Формує завдання на розробку проєкт
Менеджер проєкту	Чалий Н.	Відповідає розподіл ресурсів, завдань і термінів.

Календарний графік проєкту. Діаграма Ганта [36] є ключовим кроком в процесі планування проєкту. Ця діаграма служить як графічне відображення розкладу виконання робіт із чітким розподілом дат. Діаграма Ганта надає ясне уявлення про тривалість різних етапів проєкту, враховуючи обмеження у ресурсах, а також святкові та вихідні дні. Це важливий інструмент для ефективного керування часом та ресурсами у проєкті. Календарний графік проєкту у вигляді діаграми Ганта продемонстровано на рисунках Б.3 забезпечуючи чітке візуальне представлення термінім виконання різних фаз і завдань, що складають проєкт.

Рисунок Б.3 – Календарний графік проєкту

Управління ризиками. Управління ризиками в проєкті включає критичний аналіз та оцінку потенційних ризиків. На початковому етапі важливо провести якісну оцінку ризиків, щоб ідентифікувати ті з них, які потребують негайного вирішення. Відповідь на кожен ризик залежить від його значимості та потенційного впливу на проєкт. Після цього важливо здійснити кількісну оцінку ризиків, яка дозволяє більш точно визначити ймовірність та вплив кожного ризику. Ці оцінки можуть проводитися одночасно або окремо, залежно від рівня готовності та потреб проєкту.

У переліку ризиків проєкту, представленому в таблиці Б.3, вказані потенційні загрози та виклики. Результати оцінки цих ризиків можна знайти в таблиці Б.4. Щоб класифікувати ризики, використовується спеціальна шкала, представлена в таблиці Б.5. Ця шкала допомагає розподілити ризики за рівнем їх впливу на проєкт та ймовірністю виникнення, що є ключовим для ефективного управління ризиками.

Таблиця Б.3 – Ризики проєкту

№	Ризик	Опис ризику
1	Технічні затримки	Несподівані затримки через технічні проблеми або збої.
2	Відключення світла	Простої та втрата даних через перебої з електроенергією.
3	Недостатня якість розробки	Баги та невідповідність стандартам через слабку якість коду.
4	Проблеми зі збереженням даних	Втрата або пошкодження даних через технічні недоліки.
5	Зміни в технічних вимогах	Замовник може внести зміни у технічні вимоги під час розробки.
6	Відсутність доступу інтернету	Збої в роботі та втрата продуктивності без інтернету.
7	Проблеми сумісності з ОС	Обмеження функціональності через несумісність з ОС.
8	Кібербезпека	Несподівані затримки через технічні проблеми або збої.та бекдори
9	Затримки з боку підрядників	Затримки у роботі підрядників, наприклад, з хостингом або сторонніми сервісами.
10	Правові проблеми	Виникнення авторських або ліцензійних спорів.

Таблиця Б.4 – Результати визначення ймовірності, впливу та рангу ризиків проєкту

№	Назва (опис) ризику	Ймовірність	Вплив	Ранг
1	Технічні затримки	0.5	0.6	0.52
2	Відключення світла	0.4	0.5	0.38
3	Недостатня якість розробки	0.3	0.4	0.32
4	Проблеми зі збереженням даних	0.5	0.6	0.35
5	Зміни в технічних вимогах	0.3	0.4	0.38
6	Відсутність доступу інтернету	0.4	0.5	0.42
7	Проблеми сумісності з ОС	0.2	0.4	0.20
8	Кібербезпека	0.4	0.6	0.32
9	Затримки з боку підрядників	0.3	0.4	0.28
10	Правові проблеми	0.2	0.3	0.15

Таблиця Б.5 – Шкала оцінювання ризиків за ймовірністю виникнення та величиною впливу

Оцінка	Ймовірність виникнення	Вплив ризику	Тип ризику
1	Низька	Низький	Прийнятні
2	Середня	Середній	Виправдані
3	Висока	Високий	Недопустимі

Для мінімізації негативного впливу ризиків на проєкт необхідно ретельно спланувати стратегії реагування на них. Це включає аналіз можливих наслідків кожного ризику для проєкту та розробку відповідних планів дій. Оцінка ризиків здійснюється на основі критеріїв, зазначених у таблиці Б.4. у результаті цього планування була створена матриця ймовірності виникнення та впливу ризиків, представлена на рисунку Б.4.

У цій матриці ризики класифікуються за кольорами: зелений колір вказує на прийнятні ризики, які можуть бути легко контрольовані або мають низький вплив на проєкт, жовтий колір позначає ризики, які потребують уваги та можливо виправдані в контексті проєкту, тоді як червоний колір вказує на недопустимі ризики з високим рівнем впливу, які потребують негайного втручання. Правильне управління ризиками через цю матрицю допомагає визначити пріоритети та розробити ефективні стратегії для зменшення потенційного негативного впливу на проєкт.

Ймовірність ризику (Й)	Вплив загрози (ризику)				
	Дуже малий	Малий	Середній	Великий	Дуже велики
	0,05	0,1	0,2	0,4	0,8
0,9	0,045	0,09	0,18	0,36	0,72
0,7	0,035	0,07	0,14	0,28	0,56
0,5	0,025	0,05 R1, R8	0,10 R4	0,20	0,40
0,3	0,015 R5 R10	0,03 R6, R7	0,06 R2, R3	0,12 R10	0,24
0,1	0,005	0,01	0,02	0,04	0,08

Рисунок Б.4 – Матриця ймовірності

Класифікація ризиків проєкту за рівнем, відповідно до отриманого значення індексу, представлена у таблиці Б.6. У таблиці Б.7 описано ризики та стратегії реагування на кожен із них.

Таблиця Б.6 – Шкала оцінювання ризику за рівне

№	Назва	Межі	Ризики, які входять (номера)
1	Прийнятні	$0,005 \leq R \leq 0,05$	1,5,6,7,8,9,
2	Виправдані	$0,05 < R \leq 0,14$	2, 3, 4,10
3	Недопустимі	$0,14 < R \leq 0,72$	

Таблиця Б.7 – Ризики та стратегії реагування на них

ID	Статус ризику	Опис ризику	Ймовірність виникнення	Вплив ризику	Ранг ризику	Тип стратегії реагування	План А	План Б
1	Новий	Технічні затримки	Низька	Низький	Низький	Зменшення	Розробка детального проєктного плану з буферами часу.	Активація додаткових ресурсів для прискорення робіт.
2	Новий	Відключення світла	Середня	Середній	Середній	Ухилення	Встановлення систем неперервного живлення (UPS).	Розробка плану дій для роботи в умовах обмеженого живлення.
3	Новий	Недостатня якість розробки	Середня	Середній	Середній	Зменшення	Впровадження процесів контролю якості та регулярні ревізії.	Залучення зовнішніх консультантів для аудиту якості коду.
4	Новий	Проблеми зі збереженням даних	Середня	Середній	Середній	Зменшення	Реалізація стратегій резервного копіювання та відновлення.	Відновлення даних з резервних копій.
5	Новий	Зміни в технічних вимогах	Низька	Низький	Низький	Зменшення	Встановлення чітких каналів комунікації між стейкхолдерами.	Гнучке перепланування проєкту для адаптації до змін.
6	Новий	Відсутність доступу до мережі Інтернет	Низька	Низький	Низький	Зменшення	Підписання угоди з декількома інтернет-провайдерами.	Використання мобільного інтернету як резервного з’єднання.

Продовження табл. Б.7

ID	Статус ризику	Опис ризику	Ймовірність виникнення	Вплив ризику	Ранг ризику	Тип стратегії реагування	План А	План Б
7	Новий	Проблеми сумісності з ОС	Низька	Низький	Низький	Зменшення	Оновлення та тестування на різних ОС	Розробка патчів для сумісності
8	Новий	Кібербезпека	Низька	Низький	Низький	Зменшення	Розробка та впровадження комплексної стратегії кібербезпеки, включаючи шифрування даних, двофакторну аутентифікацію, регулярні перевірки безпеки та оновлення програмного забезпечення.	Встановлення процедур швидкого реагування на інциденти, включаючи відновлення системи та повідомлення користувачів про можливі порушення безпеки.
9	Новий	Затримки з боку підрядників	Низька	Низький	Низький	Зменшення	Подвоєння контролю термінів	Укладання договорів з штрафами
10	Новий	Правові проблеми	Середня	Середній	Середній	Зменшення	Правова експертиза контрактів	Страхування від правових ризиків

ДОДАТОК В

Лістинг коду основних компонент

OrderItemCard.tsx

```
import React from 'react';
import {
  StyleSheet,
  Text,
  View,
  ImageProps,
  Image,
  TouchableOpacity,
} from 'react-native';
import LinearGradient from 'react-native-linear-gradient';
import {
  BORDERRADIUS,
  COLORS,
  FONTFAMILY,
  FONTSIZE,
  SPACING,
} from '../theme/theme';
import CustomIcon from './CustomIcon';

interface CartItemProps {
  id: string;
  name: string;
  imagelink_square: ImageProps;
  special_ingredient: string;
  roasted: string;
```

```
  prices: any;
  type: string;
  incrementCartItemQuantityHandler: any;
  decrementCartItemQuantityHandler: any;
}

const CartItem: React.FC<CartItemProps> = ({
  id,
  name,
  imagelink_square,
  special_ingredient,
  roasted,
  prices,
  type,
  incrementCartItemQuantityHandler,
  decrementCartItemQuantityHandler,
}) => {
  return (
    <View>
      {prices.length != 1 ? (
        <LinearGradient
          start={{x: 0, y: 0}}
          end={{x: 1, y: 1}}
          colors={[COLORS.primaryGreyHex, COLORS.primaryBlackHex]}
          style={styles.CartItemLinearGradient}>
          <View style={styles.CartItemRow}>
            <Image source={imagelink_square} style={styles.CartItemImage} />
            <View style={styles.CartItemInfo}>
              <View>
                <Text style={styles.CartItemTitle}>{name}</Text>
```

```
      <Text style={styles.CartItemSubtitle}>
        {special_ingredient}
      </Text>
    </View>
    <View style={styles.CartItemRoastedContainer}>
      <Text style={styles.CartItemRoastedText}>{roasted}</Text>
    </View>
  </View>
</View>
{prices.map((data: any, index: any) => (
  <View
    key={index.toString()}
    style={styles.CartItemSizeRowContainer}>
    <View style={styles.CartItemSizeValueContainer}>
      <View style={styles.SizeBox}>
        <Text
          style={[
            styles.SizeText,
            {
              fontSize:
                type == 'Bean' ? FONTSIZE.size_12 : FONTSIZE.size_16,
            },
          ]}>
          {data.size}
        </Text>
      </View>
      <Text style={styles.SizeCurrency}>
        {data.currency}
        <Text style={styles.SizePrice}> {data.price}</Text>
      </Text>
```

```
</View>
<View style={styles.CartItemSizeValueContainer}>
 <TouchableOpacity
  style={styles.CartItemIcon}
  onPress={() => {
   decrementCartItemQuantityHandler(id, data.size);
  }}>
  <CustomIcon
   name="minus"
   color={COLORS.primaryWhiteHex}
   size={FONTSIZE.size_10}
  />
 </TouchableOpacity>
 <View style={styles.CartItemQuantityContainer}>
  <Text style={styles.CartItemQuantityText}>
   {data.quantity}
  </Text>
 </View>
 <TouchableOpacity
  style={styles.CartItemIcon}
  onPress={() => {
   incrementCartItemQuantityHandler(id, data.size);
  }}>
  <CustomIcon
   name="add"
   color={COLORS.primaryWhiteHex}
   size={FONTSIZE.size_10}
  />
 </TouchableOpacity>
</View>
```

```
      </View>
    ))}
  </LinearGradient>
) : (
  <LinearGradient
    start={{x: 0, y: 0}}
    end={{x: 1, y: 1}}
    colors={[COLORS.primaryGreyHex, COLORS.primaryBlackHex]}
    style={styles.CartItemSingleLinearGradient}>
    <View>
      <Image
        source={imagelink_square}
        style={styles.CartItemSingleImage}
      />
    </View>
    <View style={styles.CartItemSingleInfoContainer}>
      <View>
        <Text style={styles.CartItemTitle}>{name}</Text>
        <Text style={styles.CartItemSubtitle}>{special_ingredient}</Text>
      </View>
      <View style={styles.CartItemSingleSizeValueContainer}>
        <View style={styles.SizeBox}>
          <Text
            style={[
              styles.SizeText,
              {
                fontSize:
                  type == 'Bean' ? FONTSIZE.size_12 : FONTSIZE.size_16,
              },
            ]}>
```

```
      {prices[0].size}
    </Text>
  </View>
  <Text style={styles.SizeCurrency}>
    {prices[0].currency}
    <Text style={styles.SizePrice}> {prices[0].price}</Text>
  </Text>
</View>
<View style={styles.CartItemSingleQuantityContainer}>
  <TouchableOpacity
    style={styles.CartItemIcon}
    onPress={() => {
      decrementCartItemQuantityHandler(id, prices[0].size);
    }}>
    <CustomIcon
      name="minus"
      color={COLORS.primaryWhiteHex}
      size={FONTSIZE.size_10}
    />
  </TouchableOpacity>
  <View style={styles.CartItemQuantityContainer}>
    <Text style={styles.CartItemQuantityText}>
      {prices[0].quantity}
    </Text>
  </View>
  <TouchableOpacity
    style={styles.CartItemIcon}
    onPress={() => {
      incrementCartItemQuantityHandler(id, prices[0].size);
    }}>
```

```
              <CustomIcon
                name="add"
                color={COLORS.primaryWhiteHex}
                size={FONTSIZE.size_10}
              />
            </TouchableOpacity>
          </View>
        </View>
      </LinearGradient>
    )}
  </View>
 );
};

const styles = StyleSheet.create({
 CartItemLinearGradient: {
  flex: 1,
  gap: SPACING.space_12,
  padding: SPACING.space_12,
  borderRadius: BORDERRADIUS.radius_25,
 },
 CartItemRow: {
  flexDirection: 'row',
  gap: SPACING.space_12,
  flex: 1,
 },
 CartItemImage: {
  height: 130,
  width: 130,
  borderRadius: BORDERRADIUS.radius_20,
```

```
},
CartItemInfo: {
  flex: 1,
  paddingVertical: SPACING.space_4,
  justifyContent: 'space-between',
},
CartItemTitle: {
  fontFamily: FONTFAMILY.poppins_medium,
  fontSize: FONTSIZE.size_18,
  color: COLORS.primaryWhiteHex,
},
CartItemSubtitle: {
  fontFamily: FONTFAMILY.poppins_regular,
  fontSize: FONTSIZE.size_12,
  color: COLORS.secondaryLightGreyHex,
},
CartItemRoastedContainer: {
  height: 50,
  width: 50 * 2 + SPACING.space_20,
  borderRadius: BORDERRADIUS.radius_15,
  justifyContent: 'center',
  alignItems: 'center',
  backgroundColor: COLORS.primaryDarkGreyHex,
},
CartItemRoastedText: {
  fontFamily: FONTFAMILY.poppins_regular,
  fontSize: FONTSIZE.size_10,
  color: COLORS.primaryWhiteHex,
},
CartItemSizeRowContainer: {
```

```
  flex: 1,
  alignItems: 'center',
  gap: SPACING.space_20,
  flexDirection: 'row',
  justifyContent: 'center',
},
CartItemSizeValueContainer: {
  flex: 1,
  alignItems: 'center',
  flexDirection: 'row',
  justifyContent: 'space-between',
},
SizeBox: {
  backgroundColor: COLORS.primaryBlackHex,
  height: 40,
  width: 100,
  borderRadius: BORDERRADIUS.radius_10,
  justifyContent: 'center',
  alignItems: 'center',
},
SizeText: {
  fontFamily: FONTFAMILY.poppins_medium,
  color: COLORS.secondaryLightGreyHex,
},
SizeCurrency: {
  fontFamily: FONTFAMILY.poppins_semibold,
  fontSize: FONTSIZE.size_18,
  color: COLORS.primaryOrangeHex,
},
SizePrice: {
```

```
  color: COLORS.primaryWhiteHex,
},
CartItemIcon: {
  backgroundColor: COLORS.primaryOrangeHex,
  padding: SPACING.space_12,
  borderRadius: BORDERRADIUS.radius_10,
},
CartItemQuantityContainer: {
  backgroundColor: COLORS.primaryBlackHex,
  width: 80,
  borderRadius: BORDERRADIUS.radius_10,
  borderWidth: 2,
  borderColor: COLORS.primaryOrangeHex,
  alignItems: 'center',
  paddingVertical: SPACING.space_4,
},
CartItemQuantityText: {
  fontFamily: FONTFAMILY.poppins_semibold,
  fontSize: FONTSIZE.size_16,
  color: COLORS.primaryWhiteHex,
},
CartItemSingleLinearGradient: {
  flexDirection: 'row',
  alignItems: 'center',
  padding: SPACING.space_12,
  gap: SPACING.space_12,
  borderRadius: BORDERRADIUS.radius_25,
},
CartItemSingleImage: {
  height: 150,
```

```
    width: 150,
    borderRadius: BORDERRADIUS.radius_20,
  },
  CartItemSingleInfoContainer: {
    flex: 1,
    alignSelf: 'stretch',
    justifyContent: 'space-around',
  },
  CartItemSingleSizeValueContainer: {
    flexDirection: 'row',
    justifyContent: 'space-evenly',
    alignItems: 'center',
  },
  CartItemSingleQuantityContainer: {
    flexDirection: 'row',
    justifyContent: 'space-evenly',
    alignItems: 'center',
  },
});

export default CartItem;
```

jest.main.js

```
import React from 'react';
import { render } from '@testing-library/react-native';

// Імпорт компонентів
import HomeScreen from './src/screens/HomeScreen';
import DetailsScreen from './src/screens/DetailsScreen';
```

```
import CartScreen from ’./src/screens/CartScreen’;
import PaymentScreen from ’./src/screens/PaymentScreen’;

// Тести для HomeScreen
test(’renders HomeScreen correctly’, () => {
  const { getByText } = render(<HomeScreen />);
  expect(getByText(’Welcome to the Coffee Shop’)).toBeTruthy();
});

// Тести для DetailsScreen
test(’renders DetailsScreen correctly’, () => {
  const { getByText } = render(<DetailsScreen />);
  expect(getByText(’Details’)).toBeTruthy();
});

// Тести для CartScreen
test(’renders CartScreen correctly’, () => {
  const { getByText } = render(<CartScreen />);
  expect(getByText(’Your Cart’)).toBeTruthy();
});

// Тести для PaymentScreen
test(’renders PaymentScreen correctly’, () => {
  const { getByText } = render(<PaymentScreen />);
  expect(getByText(’Payment’)).toBeTruthy();
});
```