

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій
(повне найменування факультету)

Кафедра програмних засобів
(повне найменування кафедри)

Пояснювальна записка

до дипломного проекту (роботи)

бакалавр

(ступінь вищої освіти)

на тему РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ БОНУСНОЇ
СИСТЕМИ КАВ'ЯРНІ
SOFTWARE DEVELOPMENT FOR THE BONUS SYSTEM IN A CAFE

Виконав(ла): студент(ка) 4 курсу, групи КНТ-211сп
Спеціальності 122 Комп'ютерні науки
(код і найменування спеціальності)

Освітня програма (спеціалізація)

Комп'ютерні науки

ЛАШКО М.С.

(ПРИЗВИЩЕ та ініціали)

Керівник ГОЛУБ Т.В.

(ПРИЗВИЩЕ та ініціали)

Рецензент ЗЕЛЕНЬОВА І.Я.

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет КНТ
Кафедра програмних засобів
Ступінь вищої освіти бакалавр
Спеціальність 122 Комп'ютерні науки
(код і найменування)
Освітня програма (спеціалізація) Комп'ютерні науки
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
Сергій СУББОТІН
“ ” 2024 року

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

ЛАШКО Микити Сергійовича

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Розробка програмного забезпечення для бонусної системи кав'ярні. Software Development for the Bonus System in a Cafe.

керівник проєкту (роботи) к.т.н., доцент, ГОЛУБ Тетяна Василівна,
(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від “ 24 ” квітня 2024 року № 168

2. Строк подання студентом проєкту (роботи) 10 червня 2024 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області та існуючих рішень. 2. Проєктування архітектури програмного забезпечення. 3. Реалізація програмного забезпечення. 4. Тестування та впровадження.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, кількість слайдів, плакатів)

Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	ПРИЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-5 Основна частина	ГОЛУБ Т.В., доцент		
Нормоконтроль	КАМІНСЬКА Ж.К., асистент		

7. Дата видачі завдання “ 15 ” квітня 2024 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	1 тиждень	Розділ 1
3	Вибір мови програмування та інших технологій розробки.	2 тиждень	Розділ 2
4	Розробка архітектури програми.	2 тиждень	Розділ 3
5	Розробка програми.	3-4 тижні	Розділ 3, 4
6	Тестування та експериментальне дослідження програмного забезпечення.	5 тиждень	Розділ 4
7	Оформлення пояснювальної записки та документів до неї.	6 тиждень	Додатки
8	Нормоконтроль та рецензування.	7 тиждень	
9	Захист роботи.	8 тиждень	

Студент(ка)

(підпис)

Микита ЛАШКО

(Ім'я ПРИЗВИЩЕ)

Керівник проєкту (роботи)

(підпис)

Тетяна ГОЛУБ

(Ім'я ПРИЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 77 с., 13 табл., 24 рис., 3 дод., 9 джерел.

ANDROID, KOTLIN, PHP, MYSQL, МОБІЛЬНИЙ ЗАСТОСУНОК, БОНУСНА СИСТЕМА, КАВ'ЯРНЯ, ЛОЯЛЬНІСТЬ КЛІЄНТІВ.

Об'єкт дослідження – процеси управління бонусною системою кав'ярні та взаємодії з клієнтами.

Предмет дослідження – програмне забезпечення для автоматизації бонусної системи кав'ярні.

Мета роботи – розробка програмного забезпечення для ефективного управління бонусною системою кав'ярні, підвищення лояльності клієнтів та збільшення прибутковості закладу.

Матеріали, методи та технічні засоби: мова програмування Kotlin для розробки Android-додатку, PHP для серверної частини, система управління базами даних MySQL, середовище розробки Android Studio, персональний комп'ютер.

Результати. Розроблено програмне забезпечення для бонусної системи кав'ярні, яке складається з мобільного додатку для клієнтів на платформі Android, серверної частини та бази даних.

Висновки. Впровадження розробленого програмного забезпечення дозволить автоматизувати процеси управління бонусною системою кав'ярні, підвищити лояльність клієнтів за рахунок персоналізованих пропозицій та зручності використання мобільного додатку, а також збільшити прибутковість закладу завдяки стимулюванню повторних відвідувань.

Галузь використання – заклади громадського харчування, сфера обслуговування.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 77 pages, 13 tables, 24 figures, 3 appendices, 9 sources.

ANDROID, KOTLIN, PHP, MYSQL, MOBILE APP, BONUS SYSTEM, CAFE, CUSTOMER LOYALTY.

The object of the research is the management processes of the coffee shop bonus system and interaction with customers.

The subject of research is software for automating the bonus system of a coffee shop.

The goal of the work is the development of software for effective management of the coffee shop's bonus system, increasing customer loyalty and increasing the institution's profitability.

Materials, methods and technical means: Kotlin programming language for Android application development, PHP for the server part, MySQL database management system, Android Studio development environment, personal computer.

The results. The software for the bonus system of the coffee shop has been developed, which consists of a mobile application for customers on the Android platform, a server part and a database.

Conclusions. Implementation of the developed software will allow automating the management processes of the coffee shop's bonus system, increasing customer loyalty due to personalized offers and ease of use of the mobile application, as well as increasing the institution's profitability by stimulating repeat visits.

The field of use is public catering establishments, the field of service.

ЗМІСТ

	С.
Перелік скорочень та умовних познач.....	7
Вступ.....	8
1 Аналіз предметної області та існуючих рішень.....	10
1.1 Огляд бонусних систем в закладах громадського харчування	10
1.2 Аналіз переваг та недоліків існуючих рішень	12
1.3 Визначення вимог до програмного забезпечення бонусної системи кав'ярні	19
1.4 Висновки за розділом	21
2 Проектування архітектури програмного забезпечення.....	23
2.1 Опис архітектурних рішень	23
2.2 Моделювання процесів бонусної системи	24
2.3 Вибір технологій та інструментів розробки.....	27
2.4 Висновки за розділом 2	31
3 Реалізація програмного забезпечення.....	32
3.1 Опис модулів та їх функціональності.....	32
3.2 База даних	36
3.3 Інтерфейс користувача та адміністративна панель	41
3.4 Висновки за розділом 3	43
4 Тестування та впровадження	45
4.1 Стратегія та методи тестування.....	45
4.2 Результати тестування та усунення недоліків	48
4.3 Інструкції з впровадження та експлуатації системи	50
4.4 Висновки до розділу	52
Висновки	54
Перелік джерел посилань	56
Додаток А Технічне завдання	57
Додаток Б Текст програми	61
Додаток В Слайди презентації.....	71

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

- API - Application Programming;
- HTTP - Hypertext Transfer Protocol;
- QR-код - Quick Response Code;
- REST - Representational State Transfer;
- SQL - Structured Query Language;
- UI - User Interface ;
- UML - Unified Modeling Language;
- ОС - Операційна система;
- ПЗ - Програмне забезпечення;
- СКБД - Система керування базами даних.

ВСТУП

У сучасному світі, де конкуренція в бізнесі стає все більш жорсткою, компанії постійно шукають нові способи залучити та утримати лояльних клієнтів. Одним із ефективних інструментів для цього є впровадження бонусної системи, яка дозволяє винагороджувати постійних відвідувачів та заохочувати їх до подальших покупок. Кав'ярні, як і інші заклади громадського харчування, не є винятком у цій тенденції, адже задоволені клієнти є запорукою успішного бізнесу [1].

Метою даної роботи є розробка програмного забезпечення для бонусної системи кав'ярні, яке дозволить ефективно управляти процесами нарахування та використання бонусних балів клієнтами. Програмне рішення має бути зручним у використанні, надійним та масштабованим, щоб задовольнити потреби як невеликих, так і великих закладів громадського харчування.

Предметом дослідження є процеси, алгоритми та технології, необхідні для створення програмного забезпечення бонусної системи кав'ярні. Це включає аналіз вимог користувачів, розробку архітектури системи, вибір відповідних технологій та інструментів розробки, а також тестування та впровадження готового рішення.

Для досягнення поставленої мети необхідно вирішити наступні завдання:

- провести аналіз існуючих рішень та визначити переваги та недоліки різних підходів до реалізації бонусних систем;
- визначити функціональні та нефункціональні вимоги до програмного забезпечення бонусної системи кав'ярні;
- розробити архітектуру системи, яка відповідатиме вимогам масштабованості, надійності та зручності використання;
- обрати відповідні технології та інструменти розробки, які забезпечать ефективну реалізацію системи;

- реалізувати програмне забезпечення, дотримуючись принципів якісного кодування та використовуючи сучасні методології розробки;
- провести тестування програмного забезпечення на відповідність функціональним вимогам та усунути виявлені недоліки.

Для вирішення поставлених завдань будуть використані наступні методи дослідження: аналіз літературних джерел та існуючих рішень, моделювання процесів та архітектури системи, об'єктно-орієнтоване програмування, тестування програмного забезпечення, документування процесів та результатів.

Практичне значення. Розробка матиме практичне значення для власників кав'ярень та інших закладів громадського харчування, які прагнуть впровадити або вдосконалити свою бонусну систему. Розроблене програмне забезпечення дозволить їм автоматизувати процеси нарахування та використання бонусів, спростити управління бонусною програмою та підвищити задоволеність клієнтів.

Розробка програмного забезпечення для бонусної системи кав'ярні є актуальною темою, оскільки воно дозволить закладам громадського харчування ефективно управляти лояльністю своїх клієнтів та підвищувати конкурентоспроможність бізнесу. Дане дослідження охоплює всі етапи розробки програмного забезпечення, від аналізу вимог до впровадження та тестування готового рішення, і має на меті створити зручний, надійний та масштабований продукт, який задовольнить потреби як невеликих, так і великих кав'ярень.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ІСНУЮЧИХ РІШЕНЬ

1.1 Огляд бонусних систем в закладах громадського харчування

Бонусні системи стали невід'ємною частиною стратегії утримання клієнтів для багатьох закладів громадського харчування. Вони дозволяють винагороджувати постійних відвідувачів та заохочувати їх до подальших покупок, підвищуючи тим самим лояльність клієнтів та прибутковість бізнесу [2].

У закладах громадського харчування найчастіше використовуються наступні типи бонусних систем:

- програми лояльності на основі карток. Клієнти отримують спеціальні картки, на які нараховуються бонусні бали за здійснені покупки. Ці бали можна обміняти на знижки, безкоштовні товари або послуги;

- програми лояльності на основі мобільних додатків. Замість фізичних карток, клієнти використовують мобільні додатки, в яких відстежується їх активність та нараховуються бонуси. Це зручніше для клієнтів та дозволяє закладам ефективніше збирати дані про своїх відвідувачів;

- системи знижок за повторні відвідування. Клієнтам надається знижка або безкоштовний товар після певної кількості відвідувань закладу. Наприклад, після кожних десяти відвідувань клієнт отримує безкоштовну каву;

- програми, орієнтовані на події. Клієнти можуть отримувати бонуси за відвідування закладу в певні дні або під час спеціальних заходів, таких як дні народження, свята або розпродажі;

- програми партнерства. Заклади співпрацюють з іншими компаніями, пропонуючи своїм клієнтам додаткові бонуси або знижки на товари та послуги партнерів.

Незалежно від типу бонусної системи, її основна мета полягає у заохоченні клієнтів до повторних відвідувань закладу та збільшенні обсягів продажів. Успішна бонусна програма повинна бути зручною для клієнтів,

простою в управлінні для персоналу та ефективно інтегрованою в загальну стратегію маркетингу закладу (табл.1.1).

Таблиця 1.1 – Порівняння типів бонусних систем

Тип бонусної системи	Переваги	Недоліки
Програми лояльності на основі карток	- прості у використанні для клієнтів; - фізичний носій бонусів;	- необхідність виготовлення та розподілу карток; - ризик втрати або пошкодження карток;
Програми лояльності на основі мобільних додатків	- зручність для клієнтів; - збір даних про клієнтів; - легкість інформування про акції;	- не всі клієнти можуть встановити застосунок; - додаткові витрати на розробку додатку;
Системи знижок за повторні відвідування	- проста концепція для розуміння клієнтів; - мінімальні витрати на впровадження;	- обмежені можливості для персоналізації; - складність відстеження повторних відвідувань;
Програми, орієнтовані на події	- стимулювання відвідувань у певні періоди; - можливість залучення нових клієнтів;	- короткостроковий ефект - складність планування та впровадження;
Програми партнерства	- розширення переваг для клієнтів; - взаємний обмін клієнтськими базами;	- необхідність налагодження співпраці з партнерами; - ризики, пов'язані з репутацією партнерів.

Ця таблиця допомагає порівняти різні підходи до реалізації бонусних систем, розглянути їх сильні та слабкі сторони, щоб обрати оптимальний варіант або їх комбінацію для потреб кав'ярні.

Крім вищезазначених типів бонусних систем, деякі заклади громадського харчування використовують більш нестандартні підходи. Наприклад, системи на основі геолокації, де клієнти отримують бонуси за

відвідування закладу в певному місці або часі. Інший варіант - соціальні бонуси, коли заохочується обмін інформацією про заклад у соціальних мережах і рекомендації друзям.

Важливим фактором успіху бонусної програми є її простота для клієнтів. Незрозумілі чи занадто складні правила або процедури реєстрації можуть відлякати потенційних учасників. Тому більшість успішних програм мають прості та зрозумілі умови, що дозволяє легко залучати клієнтів [3].

Маркетингова складова також відіграє важливу роль. Заклади повинні постійно інформувати клієнтів про деталі бонусної програми, переваги участі, а також пропонувати спеціальні пропозиції чи знижки для учасників програми.

Вдало реалізована бонусна система може стати потужним інструментом для збільшення лояльності клієнтів та нарощування обсягів продажів. За даними досліджень, клієнти, які беруть участь у програмах лояльності, витрачають на 18-30% більше, ніж клієнти, які не є учасниками таких програм [4].

Проте слід пам'ятати, що успіх бонусної системи значною мірою залежить від її зручності та ефективності управління. Це обґрунтовує необхідність впровадження спеціалізованого програмного забезпечення, яке б автоматизувало процеси нарахування, відстеження та використання бонусів, а також надавало аналітичні дані для оптимізації програми.

1.2 Аналіз переваг та недоліків існуючих рішень

На сьогоднішньому ринку існує низка готових рішень для впровадження бонусних систем у закладах громадського харчування. Розглянемо переваги та недоліки деяких з них (табл. 1.2).

Вибір оптимального рішення залежить від розміру закладу, бюджету, наявних ресурсів та специфічних вимог до бонусної системи. Проте жодне з існуючих рішень не гарантує повного задоволення всіх потреб, що

обґрунтовує необхідність розробки спеціалізованого програмного забезпечення для бонусної системи кав'ярні [5].

Таблиця 1.2 – Готові системи для впровадження

Рішення	Переваги	Недоліки
Спеціалізовані платформи для управління лояльністю	- широкий функціонал для керування програмами лояльності; - інтеграція з різними системами лояльності; - аналітика та звітність;	- високі витрати на ліцензії та підтримку; - складність інтеграції з існуючими системами; - обмежена гнучкість налаштувань;
Модулі лояльності в POS-системах	- безпосередня інтеграція з касовими системами ; - відносно низькі витрати на впровадження;	- обмежений функціонал управління програмою ; - відсутність вебінтерфейсу для клієнтів; - складність масштабування та оновлення;
Самостійна розробка на базі фреймворків/CMS	- повна гнучкість та налаштування під потреби ; - немає прив'язки до зовнішніх постачальників ; - можливість інтеграції з різними системами;	- високі початкові витрати на розробку; - необхідність підтримки та оновлення власними силами; - ризики забезпечення належної безпеки та продуктивності.

Розглянемо приклади мобільних застосунків для трьох сучасних кав'ярень.

Оцінюючи мобільний застосунок для бонусної системи кав'ярні «КофеАрт» можна відмітити кілька аспектів, які відображають як позитивні, так і потенційно негативні сторони (рис. 1.1). За перше, яскравий та зрозумілий інтерфейс, який видно на зображенні, забезпечує простоту використання та візуальну привабливість, що є критично важливим для залучення та утримання користувачів. Прямий доступ до бонусного балансу та легкість сканування QR-коду або використання штрих-коду для отримання чи використання бонусів робить досвід користувача плавним і безшовним.

Тим не менш, розробнику слід також звернути увагу на те, що застосунок має бути масштабованим і безпечним, з врахуванням можливих ризиків безпеки, пов'язаних з транзакціями та зберіганням особистих даних користувачів. Потрібно впевнитися, що система захищена від шахрайства та зловмисних атак. Крім того, важливо забезпечити інтеграцію з іншими системами кав'ярні, наприклад з системою обліку та касовими апаратами, що може становити технічний виклик [6].

Основна проблема для розробника полягає у забезпеченні стабільності роботи додатку під час піків користувацької активності, оскільки велика кількість одночасних запитів може призвести до збоїв або сповільнення роботи системи. Крім того, необхідно забезпечити високий рівень користувацького досвіду на різних пристроях і платформах, зважаючи на різноманітність пристроїв та версій операційних систем, які можуть використовувати користувачі.

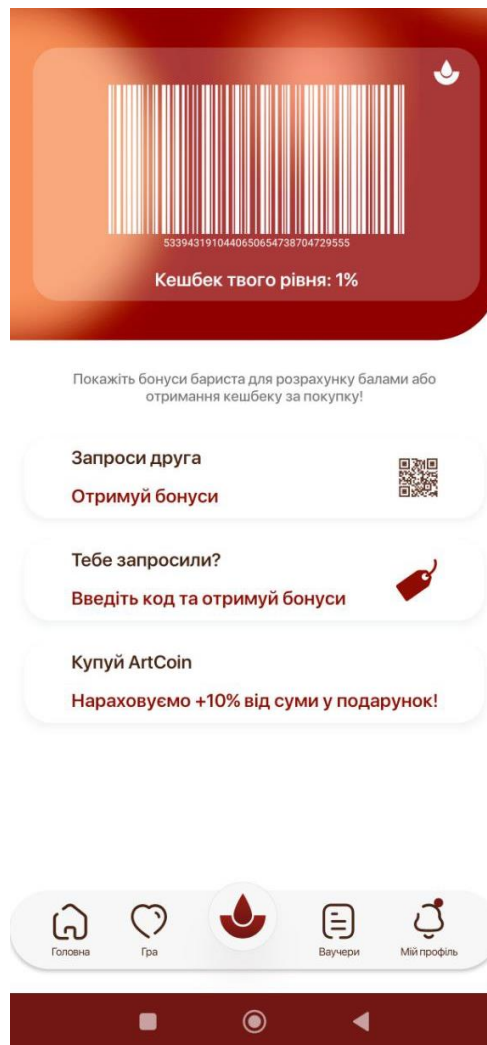


Рисунок 1.1 – Кав'ярня «КофеАрт» [7]

Оглядаючи застосунок для бонусної системи кав'ярні «Лакі Кофе» на рисунку 1.2 можна помітити, що дизайн є чітким та користуваче-орієнтованим, що сприяє легкій навігації і зрозумілості для користувачів. Елементи брендування, такі як логотип і фірмові кольори, використовуються послідовно, що підсилює впізнаваність кав'ярні. Центральне розміщення балів лояльності відразу привертає увагу, що є ключовим для мотивації користувачів використовувати застосунок регулярно.

Інтеграція з картою для вибору локації є корисною функцією, яка може полегшити користувачам знаходження найближчої кав'ярні, але це також вимагає постійного підтримання точності геолокаційних даних та високої якості карти, щоб забезпечити гладкий досвід користувача.



Рисунок 1.2 – Кав'ярня «Лакі Кофе» [8]

З іншого боку, мобільний застосунок може мати потенційні недоліки залежно від його функціональності та масштабованості. Наприклад, якщо застосунок не оптимізовано для різних пристроїв, це може призвести до проблем з користувачем на менших екранах або на пристроях з різними операційними системами. Якщо інфраструктура, яка підтримує застосунок, не є достатньо міцною, це може призвести до технічних збоїв або повільної роботи додатку під час пікових навантажень, що може відштовхнути користувачів.

Безпека даних є ще одним критичним аспектом; якщо застосунок не використовує надійні методи шифрування та безпечного зберігання особистої інформації користувачів, це може призвести до витоку даних та інших проблем з безпекою. Також необхідно враховувати можливість швидкого оновлення додатку для відповіді на змінювані вимоги користувачів та бізнесу, що може бути викликом у плані управління проектом і розподілу ресурсів.

У підсумку, застосунок здається привабливим та зручним для користувача, з потенціалом для розвитку лояльності та залучення клієнтів, але як розробник повинен звернути увагу на технічне втілення, безпеку, масштабованість і постійне оновлення, щоб забезпечити стабільність і довгостроковий успіх [3].

Розглядаючи скріншот мобільного додатку для бонусної системи кав'ярні «Кофе Лат», можна висвітлити кілька моментів, що впливають на його оцінку (рис. 1.3). Цей застосунок виглядає функціональним і зручним, з легкодоступними кнопками для навігації, як "Карта" та "Сканер QR", що спрощує взаємодію з користувачем. Використання яскравих кольорів та великих кнопок робить інтерфейс привабливим і зрозумілим для користувачів, що сприяє підвищенню їх залученості. Розділ "Новини та акції" дає можливість кав'ярні оперативно інформувати клієнтів про спеціальні пропозиції та залучати їх до частішого відвідування закладу, що є важливою перевагою в контексті маркетингу та збільшення продажів.

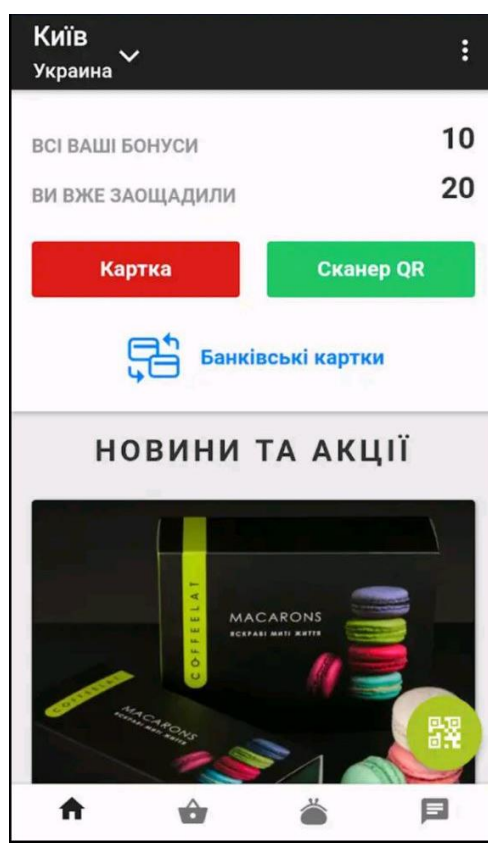


Рисунок 1.3 – Кав'ярня «Кофе Лат» [9]

З іншої сторони, застосунок має чітко відображати інформацію про бонусні бали та рівень досягнень, щоб користувачі могли легко розуміти, які переваги вони отримують, використовуючи застосунок. Якщо інтерфейс не надає достатньо інформації або робить це нечітко, це може призвести до непорозумінь та незадоволення з боку користувачів.

Наведемо порівняльну таблицю додатків (табл. 1.3).

Таблиця 1.3 – Порівняння застосунків

Застосунок	Переваги	Недоліки
Кофе Арт	Інтуїтивний інтерфейс, яскравий дизайн, простота використання, візуальна привабливість, легке сканування QR.	Потреба в масштабованості та безпеці даних, ризик збоїв при великій кількості запитів.
Лакі Кофе	Чистий дизайн, легка навігація, послідовне брендуння, інтеграція з картою локацій.	Потенційні проблеми з користувачем на менших екранах, важливість стабільної роботи додатку під час пікових навантажень.
Кофе Лат	Функціональність, зручні кнопки для навігації, яскраві кольори, розділ "Новини та акції" для маркетингу.	Необхідність чіткого відображення інформації про бонуси, висока важливість безпеки даних та продуктивності.

З технічної точки зору, важливо забезпечити, щоб застосунок працював стабільно та ефективно на різних пристроях і операційних системах, а також мав високу продуктивність навіть при великому обсязі користувацьких даних та запитів. Розробникам слід також враховувати безпеку даних користувачів, оскільки застосунок займається обробкою

особистої інформації та платіжних даних. Недостатня увага до аспектів безпеки може призвести до витоку даних, що матиме негативні наслідки для репутації кав'ярні та довіри користувачів.

У підсумку, застосунок має великий потенціал для підвищення лояльності клієнтів та зростання бізнесу, але розробник повинен уважно ставитися до дизайну інтерфейсу, продуктивності, масштабованості та безпеки, щоб забезпечити успішне та тривале використання додатку.

1.3 Визначення вимог до програмного забезпечення бонусної системи кав'ярні

Ключовим рішенням для розробки програмного забезпечення бонусної системи кав'ярні буде створення мобільного додатку. Це забезпечить максимальну зручність для клієнтів, оскільки вони зможуть легко перевіряти свій бонусний рахунок, відстежувати нараховані бали та отримувати інформацію про спеціальні пропозиції прямо зі свого смартфона. Крім того, мобільний застосунок дозволить збирати цінні дані про клієнтів, їхні вподобання та поведінку, що допоможе кав'ярні ефективніше налаштовувати маркетингові кампанії та покращувати якість обслуговування [2]. Таблиця 1.4 описує етапи вибору технологій та інструментів для розробки мобільного додатку бонусної системи кав'ярні.

Основні функціональні вимоги до програмного забезпечення включають можливість реєстрації та авторизації користувачів, перегляд бонусного рахунку, відстеження історії транзакцій, інформування про нові акції та пропозиції, а також зручний інтерфейс для використання бонусних балів під час оплати в кав'ярні.

Таблиця 1.4 – Етапи розробки програмного забезпечення бонусної системи кав'ярні

№	Назва етапу	Опис
1	Визначення вимог до додатку	Чітке уявлення функціоналу, який повинен реалізувати застосунок.
2	Аналіз цільової аудиторії	Розуміння використання додатку кінцевими користувачами.
3	Вибір платформи	Рішення про розробку нативного додатку під конкретну платформу (iOS або Android) або кросплатформного додатку, що працюватиме на обох платформах, значно вплине на вибір інструментів та мов програмування.
4	Вибір мови програмування та інструментарію	Залежно від обраної платформи, розробники мають вирішити, чи використовувати Swift або Objective-C для iOS, Java або Kotlin для Android, або ж кросплатформні рішення як Flutter або React Native.
5	Вибір архітектури додатку	Важливо визначити архітектуру додатку (наприклад, MVC, MVP, MVVM), яка забезпечить його масштабованість, підтримку та легкість в додаванні нових функцій.
6	Інтеграція з зовнішніми сервісами та API	Бонусна система, швидше за все, вимагатиме інтеграції з базою даних, системами управління замовленнями, платіжними системами та можливо з соціальними мережами.
7	Вибір інструментів для тестування та розгортання	Використання засобів для контролю якості додатку та його доставки користувачам.
8	Забезпечення безпеки	Застосування заходів для захисту даних користувачів та транзакцій.
9	Врахування майбутньої підтримки та масштабування	Планування легкості внесення змін та обробки збільшеного навантаження.

Нефункціональні вимоги передбачають, що програмне забезпечення має бути безпечним, надійним, масштабованим та сумісним із різними

мобільними платформами. Важливими аспектами є також швидкодія та зручність використання для забезпечення позитивного досвіду користувачів.

Інтеграція з існуючими системами кав'ярні, такими як POS-термінали та системи управління запасами, є ключовою вимогою для безперебійної роботи бонусної програми та ефективного обміну даними між різними компонентами. Дані про покупки клієнтів та нараховані бонуси повинні синхронізуватися в режимі реального часу, щоб забезпечити актуальність інформації в мобільному додатку та уникнути можливих конфліктів.

Крім того, програмне забезпечення має забезпечувати зручний спосіб адміністрування бонусної програми для персоналу кав'ярні, включаючи можливість створення та редагування акцій, налаштування правил нарахування бонусів та генерацію звітів про ефективність програми.

1.4 Висновки за розділом 1

У розділі було проведено ґрунтовний аналіз предметної області та існуючих рішень для впровадження бонусних систем у закладах громадського харчування, зокрема в кав'ярнях. Було розглянуто різні типи бонусних систем, їхні переваги та недоліки, а також фактори, що впливають на успіх таких програм.

Дослідження показало, що на ринку присутні різноманітні готові рішення, такі як спеціалізовані платформи, модулі в POS-системах та можливість самостійної розробки. Проте жодне з цих рішень не є ідеальним і не задовольняє усі потреби кав'ярень повною мірою. Тому було обґрунтовано необхідність розробки спеціалізованого програмного забезпечення для бонусної системи.

Ключовим рішенням визначено створення мобільного додатку, який забезпечить максимальну зручність для клієнтів, можливість збору даних про їхню поведінку та ефективну інтеграцію з існуючими системами кав'ярні. Було сформульовано основні функціональні та нефункціональні вимоги до

такого програмного забезпечення, включаючи безпеку, надійність, масштабованість, швидкодію та зручність використання.

Загалом, розділ підкреслив важливість впровадження ефективної бонусної системи для підвищення лояльності клієнтів та зростання бізнесу кав'ярні. Розробка спеціалізованого програмного забезпечення з урахуванням визначених вимог є перспективним шляхом для досягнення цієї мети.

2 ПРОЕКТУВАННЯ АРХІТЕКТУРИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Опис архітектурних рішень

Для забезпечення ефективної роботи бонусної системи кав'ярні необхідно розробити гнучку та масштабовану архітектуру програмного забезпечення [3]. У цьому контексті доцільно використати клієнт-серверну архітектуру з трьохрівневою структурою (рис. 2.1).

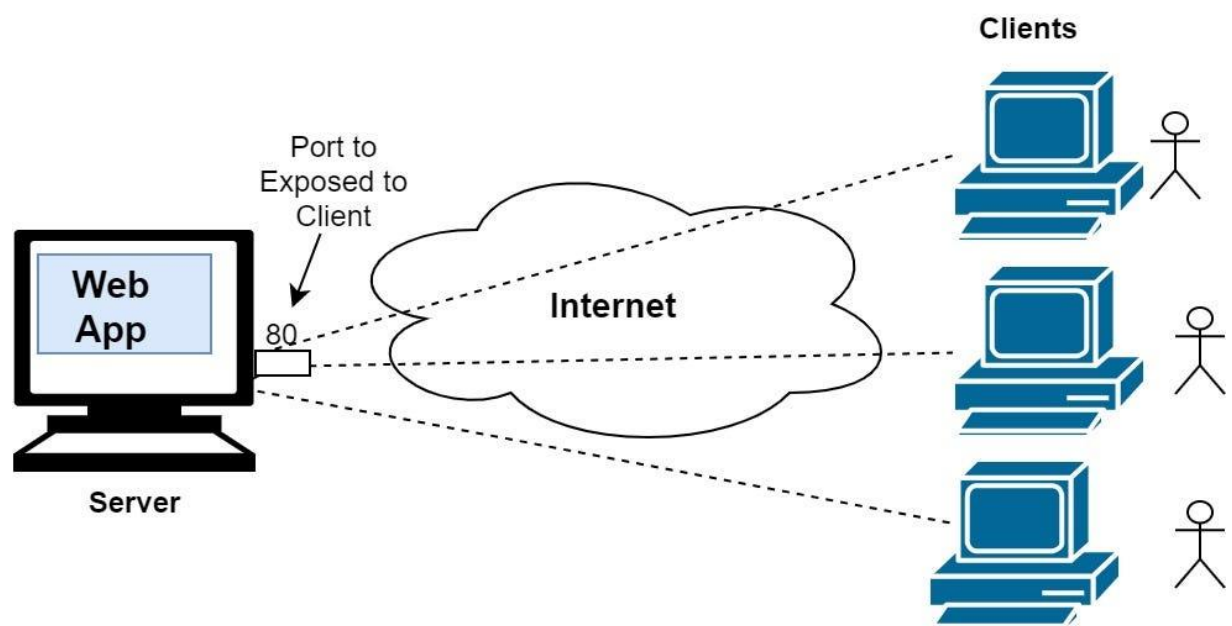


Рисунок 2.1 - Клієнт-серверну архітектура з трьохрівневою структурою

Клієнтський рівень представлений мобільним додатком, який буде доступний для користувачів на платформах iOS та Android. Цей застосунок забезпечить зручний інтерфейс для перегляду бонусного рахунку, історії транзакцій, інформації про акції та використання накопичених балів під час оплати в кав'ярні [1].

Серверний рівень складатиметься з двох основних компонентів: вебсервера та бази даних. Вебсервер буде відповідати за обробку запитів від мобільного додатку, взаємодію з базою даних та іншими системами кав'ярні, а також за реалізацію бізнес-логіки бонусної програми [5].

База даних зберігатиме інформацію про користувачів, їхні бонусні рахунки, транзакції, акції та інші дані, необхідні для функціонування системи. Для забезпечення високої продуктивності та масштабованості доцільно використати реляційну базу даних, наприклад, PostgreSQL або MySQL.

Взаємодія між клієнтським і серверним рівнями відбуватиметься через RESTful API, що забезпечить сумісність з різними платформами та спростить інтеграцію з іншими системами кав'ярні [7].

Для підвищення безпеки та конфіденційності даних буде впроваджено шифрування під час передачі даних між клієнтом і сервером, а також використано сучасні методи аутентифікації та авторизації користувачів.

Крім того, архітектура передбачатиме окремий адміністративний вебінтерфейс для керування бонусною системою персоналом кав'ярні. Цей інтерфейс дозволить створювати та редагувати акції, налаштовувати правила нарахування бонусів, генерувати звіти про ефективність програми та виконувати інші адміністративні завдання [5].

Для забезпечення високої доступності та відмовостійкості система буде розгорнута на кількох серверах з використанням технологій балансування навантаження та кешування даних.

Ця архітектура дозволить створити масштабовану, безпечну та зручну у використанні бонусну систему для кав'ярні, яка задовольнить потреби як клієнтів, так і персоналу закладу.

2.2 Моделювання процесів бонусної системи

Для забезпечення ефективної розробки та подальшого впровадження програмного забезпечення бонусної системи кав'ярні важливо детально змодельовати всі ключові процеси та потоки даних. Це дозволить виявити потенційні проблеми та оптимізувати систему на ранніх етапах.

Одним із основних процесів є реєстрація нового користувача в бонусній програмі. Цей процес можна представити за допомогою діаграми (рис. 2.2).

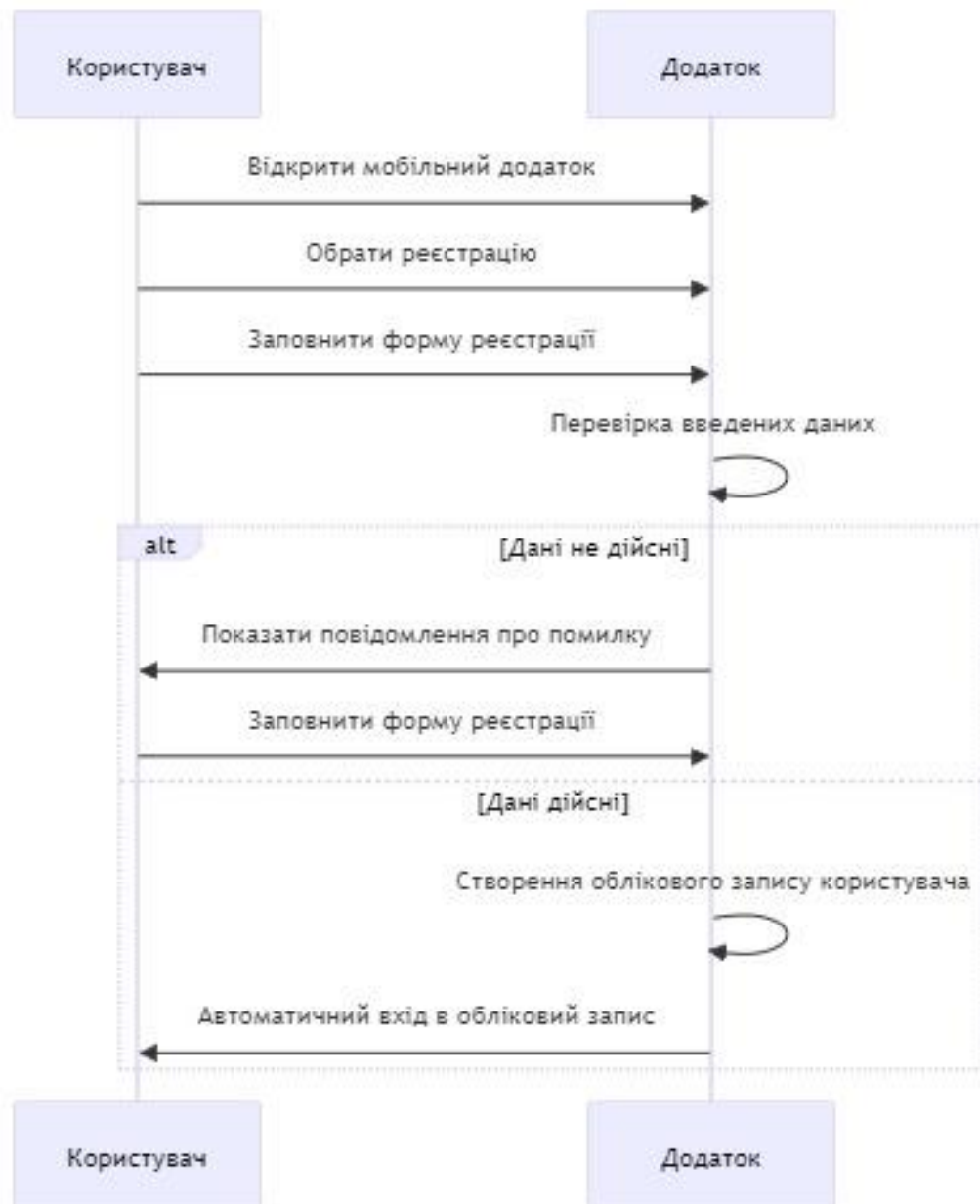


Рисунок 2.2 – Діаграма послідовності реєстрації нового користувача

Дана діаграма показує, що спочатку користувач відкриває мобільний застосунок і обирає функцію реєстрації. Далі він заповнює реєстраційну форму, після чого система валідує введені дані. Якщо дані некоректні, користувачеві виводиться повідомлення про помилку. Якщо ж дані валідні,

система створює новий акаунт користувача в базі даних, після чого користувач може автоматично увійти до створеного акаунту.

Іншим ключовим процесом є нарахування та використання бонусних балів. Діаграма (рис. 2.3) демонструє взаємодію між різними компонентами системи під час цього процесу.

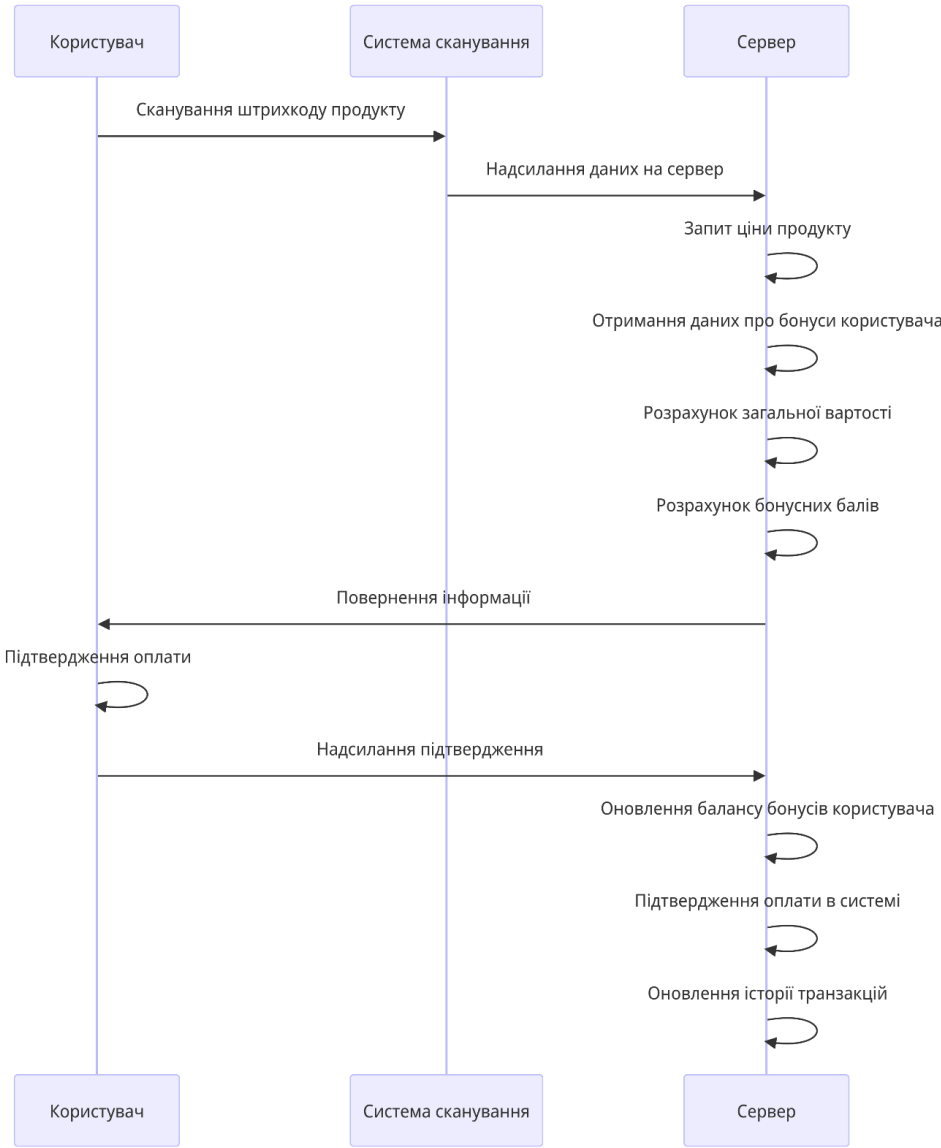


Рисунок 2.3 – Діаграма послідовності нарахування бонусів

На діаграмі видно, що початковим кроком є сканування штрих-коду продукту користувачем в мобільному додатку. Застосунок надсилає ці дані на сервер. Сервер, у свою чергу, запитує ціну продукту з бази даних, а також дані про бонусний рахунок користувача. На основі цих даних сервер

розраховує загальну вартість покупки та кількість бонусних балів, які буде нараховано, і повертає цю інформацію у застосунок. Після підтвердження платежу користувачем застосунок надсилає підтвердження на сервер. Сервер оновлює бонусний баланс користувача в базі даних, підтверджує успішну оплату в касовій системі кав'ярні та оновлює історію транзакцій у базі даних.

Моделювання ключових процесів дозволяє виявити потенційні вузькі місця, визначити необхідні інтеграції та спростити послідовну розробку програмного забезпечення бонусної системи.

2.3 Вибір технологій та інструментів розробки

Вибір відповідних технологій та інструментів розробки відіграє ключову роль для забезпечення ефективності, надійності та масштабованості програмного забезпечення бонусної системи кав'ярні. В цьому розділі розглянемо технології, які будуть використані для реалізації різних компонентів системи [1].

Порівнюючи додатки iOS та Android, ми бачимо, що вони розробляються в різних середовищах: iOS додатки використовують Swift або Objective-C і розробляються в Xcode, тоді як Android додатки зазвичай пишуть на Java або Kotlin і використовують Android Studio. У додатках iOS зазвичай є більш строгі вимоги до дизайну інтерфейсу згідно з Apple Human Interface Guidelines, що призводить до більшої уніфікації та часто вищої якості дизайну, у той час як Google Material Design дає більше гнучкості для Android додатків, але це також може спричинити більшу фрагментацію вигляду та відчуття додатків (табл. 2.1).

Через відкритий характер Android, розробники можуть мати більше можливостей для інтеграції з системними можливостями і сторонніми сервісами, тоді як iOS надає більш обмежений доступ з міркувань безпеки та приватності. Це також позначається на розповсюдженні додатків: Apple App Store має суворішу політику перевірки, що може забезпечити більшу якість і

безпеку, але також створює додаткові бар'єри для розробників. Google Play має менш жорсткі правила, що може призвести до швидшого випуску додатків, але також збільшує ризик появи шкідливих або низькоякісних додатків [2].

Таблиця 2.1 – Порівняння додатків iOS та Android

Аспект	iOS	Android
Мови програмування та інструменти	Swift або Objective-C, інтегроване середовище розробки Xcode	Java або Kotlin, інтегроване середовище розробки Android Studio
Доступ до системи та налаштування	Обмежений доступ до системи для підвищення безпеки	Відкритіший доступ, більш широкі можливості налаштування
Вимоги до дизайну інтерфейсу	Строге дотримання вказівок Apple Human Interface Guidelines	Гнучкі вказівки Google Material Design
Вимоги до публікації в магазині	Суворий процес перевірки в App Store	Більш м'який процес перевірки в Google Play
Заходи безпеки	Високі, завдяки суворому контролю за додатками та конфіденційністю	Можуть бути менші через відкриту природу екосистеми
Вартість розробки	Зазвичай вища через ретельне тестування та контроль якості	Зазвичай нижча через широкий спектр пристроїв та менш жорсткі вимоги
Частка ринку	Менша у порівнянні з Android, але користувачі частіше готові платити	Більша у порівнянні з iOS, з широким спектром пристроїв та цінових точок

Щодо ринку, iOS користувачі зазвичай вважаються більш вигідною аудиторією для монетизації через додатки, тому що вони схильні витратити більше грошей на покупки в додатках і підписки, у той час як аудиторія Android є більшою за кількістю, але менш схильною до витрат в додатках.

При розробці мобільного додатку важливо враховувати особливості платформи, для якої створюється застосунок. Для iOS найпопулярнішими

мовами програмування є Swift та Objective-C. Swift – це більш нова мова з простим і виразним синтаксисом, яка спрощує розробку і зменшує потенціал для помилок. Objective-C – це старша мова, яка все ще використовується для підтримки існуючих додатків або у випадках, коли необхідна висока сумісність з давно розробленими бібліотеками [5].

Для Android найбільш розповсюдженими мовами є Java та Kotlin. Java була домінантною мовою для розробки на Android протягом багатьох років і має велику базу розробників та ресурсів. Однак, Kotlin швидко набув популярності після того, як був оголошений Google як офіційна мова розробки для Android у 2017 році. Kotlin пропонує більш сучасний і зручний синтаксис, кращу сумісність з Java і зменшений потенціал для помилок завдяки нуль-безпечним типам і імутабельності.

Крім нативної розробки, існують кросплатформні рішення, які дозволяють розробникам писати код один раз і запускати його на обох платформах. Найвідомішими інструментами кросплатформної розробки є Flutter і React Native. Flutter використовує мову Dart і надає широкий набір віджетів для створення графічного інтерфейсу. React Native базується на JavaScript і дозволяє розробникам використовувати React разом з нативними можливостями платформ (табл. 2.2).

Вибір Kotlin для розробки Android-додатків обґрунтований кількома перевагами:

- сучасний синтаксис. Kotlin пропонує чистий та виразний синтаксис, який робить код легшим для читання та написання;
- безпека. Kotlin зменшує ймовірність поширених помилок, таких як null pointer exceptions, завдяки системі типів, яка включає обробку нульових значень;
- сумісність з Java. Kotlin повністю сумісний з Java, що дозволяє легко інтегрувати Kotlin у існуючі проекти на Java без необхідності переписування всього коду;

- підтримка спільноти та Google. З оголошенням Kotlin офіційною мовою розробки Android, Google забезпечує потужну підтримку мови, а спільнота Kotlin швидко розвивається.

Таблиця 2.2 – Порівняння Java та Kotlin

Критерій	Java	Kotlin
Мова програмування	Старша, широко використовується для Android-розробки	Новіша, офіційно підтримується Google для Android-розробки
Синтаксис	Більш об'ємний і складний	Коротший і чистіший, спрощує читання та написання коду
Безпека використання null	Частіше стикається з помилками NullPointerException	Використовує безпечні від null типи, щоб уникнути NullPointerException
Сумісність з Java	Нативна сумісність, оскільки Kotlin може використовувати Java бібліотеки і навпаки	Повна сумісність, дозволяє легко інтегрувати Kotlin у існуючі Java проекти
Функціональне програмування	Обмежені можливості порівняно з Kotlin	Підтримує функціональні парадигми ефективніше, ніж Java
Корутини	Не підтримується нативно, потрібні додаткові бібліотеки	Підтримується нативно, спрощує асинхронне програмування
Спільнота та підтримка	Велика та досвідчена, але Kotlin набуває популярності	Швидко зростаюча з сильною підтримкою від Google і спільноти

Завдяки цим перевагам, Kotlin є відмінним вибором для розробки сучасних, безпечних і легко підтримуваних Android-додатків.

2.4 Висновки за розділом 2

Даний розділ описує архітектуру, моделювання процесів та вибір технологій для розробки масштабованої та безпечної бонусної системи кав'ярні.

Для реалізації бонусної системи кав'ярні пропонується використати клієнт-серверну архітектуру із трирівневою структурою, що включає мобільний застосунок для клієнтів, вебсервер та базу даних на серверному рівні. Мобільний застосунок буде розроблений для платформи Android, забезпечуючи зручний інтерфейс для перегляду бонусного рахунку, історії транзакцій та використання балів під час оплати. Взаємодія між клієнтом і сервером відбуватиметься через RESTful API, що забезпечить сумісність та інтеграцію з іншими системами кав'ярні.

Для моделювання процесів використовуються діаграми UML, такі як діаграма діяльності для реєстрації користувачів та діаграма послідовності для нарахування бонусів. Для розробки мобільного додатку на Android рекомендується використовувати мову програмування Kotlin, яка пропонує сучасний синтаксис, підвищену безпеку, сумісність з Java та підтримку Google і спільноти.

3 РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Опис модулів та їх функціональності

Для операційної системи Android розроблено програмний застосунок, який складається з кількох активностей, кожна з яких пов'язана з візуальною частиною застосунку. Компонент Activity є ключовим елементом додатку на платформі Android, а процес ініціації та взаємодії активностей лежить в основі архітектури застосунків даної платформи. Відрізняючись від традиційних програмних парадигм, де використовується метод `main()` для запуску програми, Android стартує застосунок через екземпляр Activity, активуючи специфічні callback-методи, що відповідають за різні стадії життєвого циклу активності. Кожна активність в проекті представлена класом, написаним на Kotlin, що міститься у файлі з розширенням `.kt` (рис. 3.1).

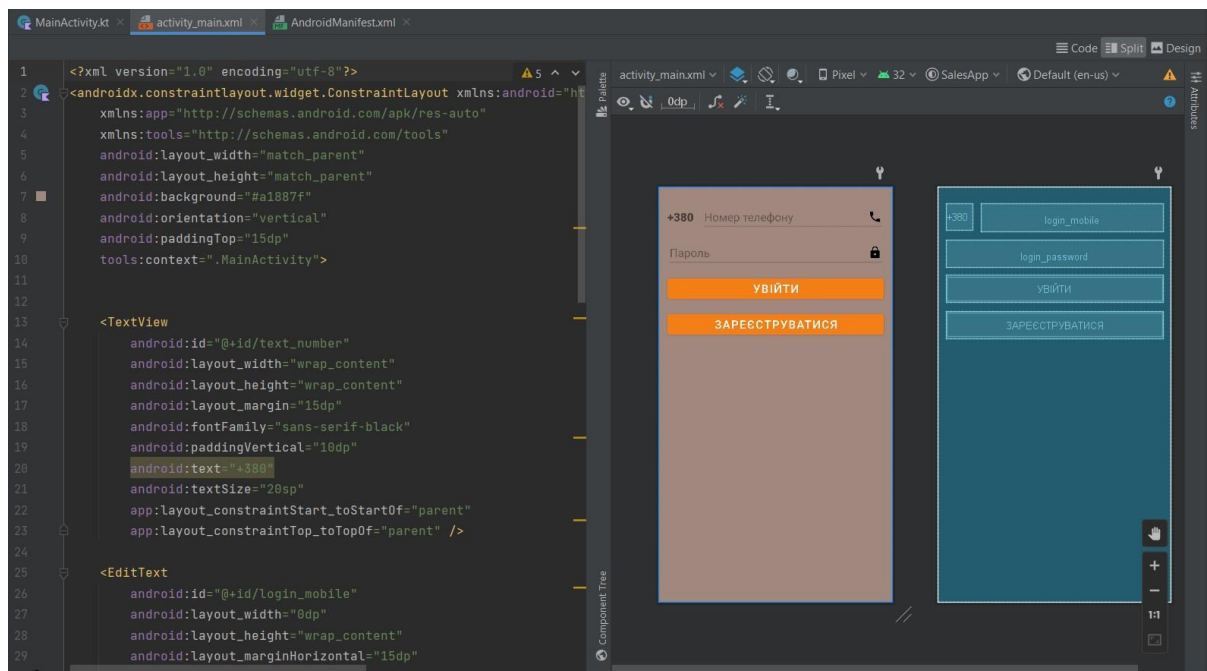


Рисунок 3.1 - Застосунок

Для кожної активності існує відповідний xml-файл, який містить xml-код для визначення розміщення візуальних компонентів застосунку.

Запускаючи активність, система Android адаптує візуальний контент до розмірів екрану пристрою на основі опису в xml-файлі, забезпечуючи консистентність відображення активності на різних пристроях.

Програмний застосунок також включає в себе файл `AndroidManifest.xml`, розташований у кореневій директорії проекту, який містить критично важливі відомості про застосунок для збіркових інструментів Android, операційної системи та Google Play.

При створенні нового проекту в Android Studio, ці компоненти генеруються автоматично з базовими налаштуваннями, але їх потрібно доповнити або модифікувати відповідно до специфіки розроблюваного додатку, а також додавати нові активності відповідно до вимог. У файлі `AndroidManifest.xml` більшість елементів додається в процесі розробки, особливо при використанні шаблонів коду.

Розроблений застосунок взаємодіє з віртуальним сервером, тому в основу проекту покладено "клієнт-серверну архітектуру". У цій моделі, мобільний застосунок (клієнт) надсилає запити на сервер для обробки даних або виконання системних дій. Сервер обробляє запити та повертає результати, здатний одночасно обслуговувати кілька клієнтів. У проекті використано WampServer як серверну платформу, що дозволяє розробляти вебдодатки на Windows, використовуючи Apache, PHP, MySQL та MariaDB. WampServer функціонує як локальний сервер на комп'ютері розробника, що спрощує тестування та розробку проекту без необхідності підключення до Інтернету. Віртуальний хостинг та домен для сервера були налаштовані для забезпечення доступу до додатку не лише в локальній мережі.

У цьому застосунку реалізовано кілька ключових компонентів (модулів), які дозволяють взаємодіяти в межах Android-середовища. Застосунок містить декілька активностей, між якими можна переключатися в процесі його використання (рис. 3.2).

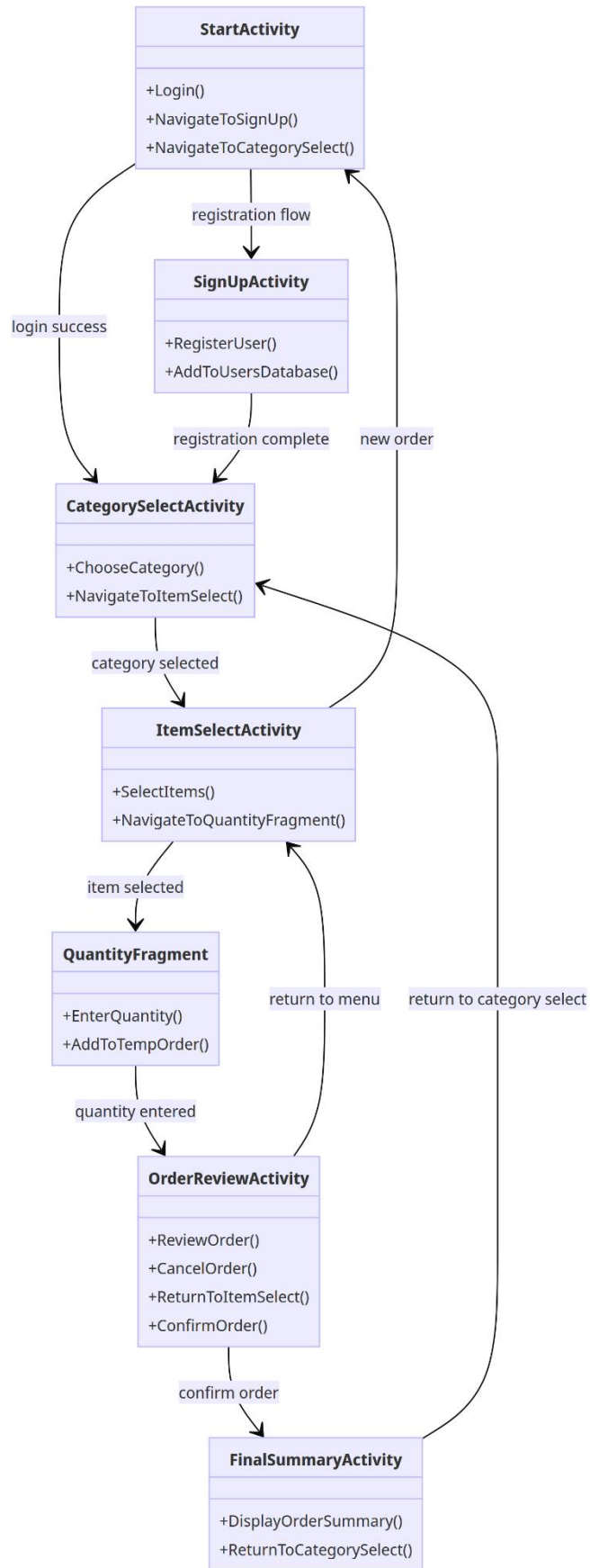


Рисунок 3.2 – Діаграма класів додатку

StartActivity – основна активність, що включає в себе два поля для текстового вводу та дві кнопки. Ця активність призначена для ідентифікації користувача та навігації до модулів реєстрації (SignUpActivity) та вибору категорій (CategorySelectActivity). Під час авторизації здійснюється перевірка введених даних з використанням скрипта "login.php", що перевіряє присутність користувача в базі даних "users".

SignUpActivity – модуль реєстрації, де користувач може зареєструватися, вносячи свої дані до бази даних. Тут перевіряється валідність даних, після чого скрипт "add_user.php" додає інформацію користувача до таблиці "users".

CategorySelectActivity – активність для вибору категорій, яка представляє список категорій, з якого користувач може вибрати потрібну для подальшого уточнення товарів. Вибравши категорію, відбувається перехід до наступної активності, де передається інформація про обрані категорії. Використовується ListView для форматування переліку. Дані з бази даних витягуються через скрипт "get_cat.php" з таблиці "items".

ItemSelectActivity – активність, де відбувається вибір конкретних товарів відповідно до попередньо вибраної категорії. Отримавши дані про обрані товари, активність формує список товарів за допомогою адаптера "ItemAdapter", взаємодіючи з базою даних через скрипт "get_items.php", який зчитує вміст таблиці "items".

QuantityFragment – фрагмент, що активується після вибору товару та містить поля для введення кількості обраного товару. Дозволяє користувачу вказати кількість товару та передає ці дані до бази даних через скрипт "add_temp.php", формуючи тимчасове замовлення у таблиці "temp_order".

OrderReviewActivity – модуль, який отримує інформацію про тимчасове замовлення з бази даних за допомогою скрипта "get_temp.php" та відображає її у списку. Також містить меню з опціями для подальших дій. Кнопка "Скасувати" ініціює скрипт "cancel_order.php", який видаляє тимчасове замовлення. При виборі "Повернутися до меню", відбувається

перехід назад до `CategorySelectActivity`, дозволяючи додати нові товари. "Підтвердити" запускає `"confirm_order.php"`, реєструючи остаточне замовлення в базі даних та видаляючи тимчасове замовлення.

`FinalSummaryActivity` – остання активність, що підсумовує та відображає загальну вартість замовлення разом з детальною інформацією. Включає кнопку для повернення до вибору категорій (`CategorySelectActivity`).

Кожен PHP-скрипт виконує свої специфічні завдання, використовуючи SQL-запити для обміну даними з базою даних, та повертає результати до відповідної активності.

3.2 База даних

Програмний застосунок взаємодіє з реляційною базою даних MySQL, яка є відомим сервером баз даних, часто включеним до комплектів різноманітних програмних продуктів. SQL, що у назві MySQL, відноситься до мови структурованих запитів, яка використовується для взаємодії з базами даних. MySQL надає додаткові можливості мови SQL для розширення функціональності користувачів.

Адміністрування бази даних MySQL відбувається за допомогою `phpMyAdmin`, графічного інтерфейсу веббраузера, який дозволяє керувати сервером MySQL, зокрема виконувати SQL-запити та керувати вмістом таблиць.

Скрипти для обробки клієнтських запитів до сервера розроблено в Apache NetBeans на мові PHP. NetBeans є відкритим інтегрованим середовищем розробки для різних мов програмування, включаючи PHP, JavaScript, HTML5, і підтримується на багатьох платформах, таких як Windows, Linux, FreeBSD, Solaris. NetBeans забезпечує широкий спектр можливостей для веброзробки, включаючи рефакторинг коду, підсвічування синтаксису, профілювання додатків, автодоповнення та шаблони коду.PHP

використовується для написання серверних скриптів, що дозволяє генерувати динамічні вебсторінки, обробляти дані форм і керувати запитами.

Вибір бази даних є ключовим етапом розробки, оскільки обране рішення повинно відповідати як поточним, так і майбутнім вимогам програми. MySQL використовується як для малих, так і для середніх застосунків і часто служить вебсервером для локальних та віддалених клієнтів. MySQL включає бібліотеку внутрішнього сервера, що дозволяє інтегрувати MySQL у автономні програми, наприклад, у вебсервер WAMP.

MySQL, з його відкритою архітектурою і GPL-ліцензуванням, продовжує розширювати свої можливості з появою нових типів таблиць, таких як MyISAM, що підтримують повнотекстовий пошук, і InnoDB, що підтримують транзакції на рівні окремих записів. Переваги MySQL включають підтримку множинних запитів, багатопоточність, оптимізацію зв'язків для швидкого об'єднання даних, спеціальні і ключові поля, гнучку систему привілеїв, швидку систему пам'яті на основі потоків, підтримку рядків різної довжини і міток часу, просте управління таблицями. Важливо, що операції з рядками в MySQL не залежать від регістру символів.

Розроблена база даних базується на реляційній моделі і була спроектована з урахуванням аналізу сутностей та функцій додатку для кав'ярні. Проведена нормалізація даних та визначені ключові поля. Схема бази даних бонусної програми для Android ілюструє структуру таблиць та їх взаємозв'язки, забезпечуючи ефективне управління даними через SQL-запити, створені в PHP скриптах (рис. 3.3).

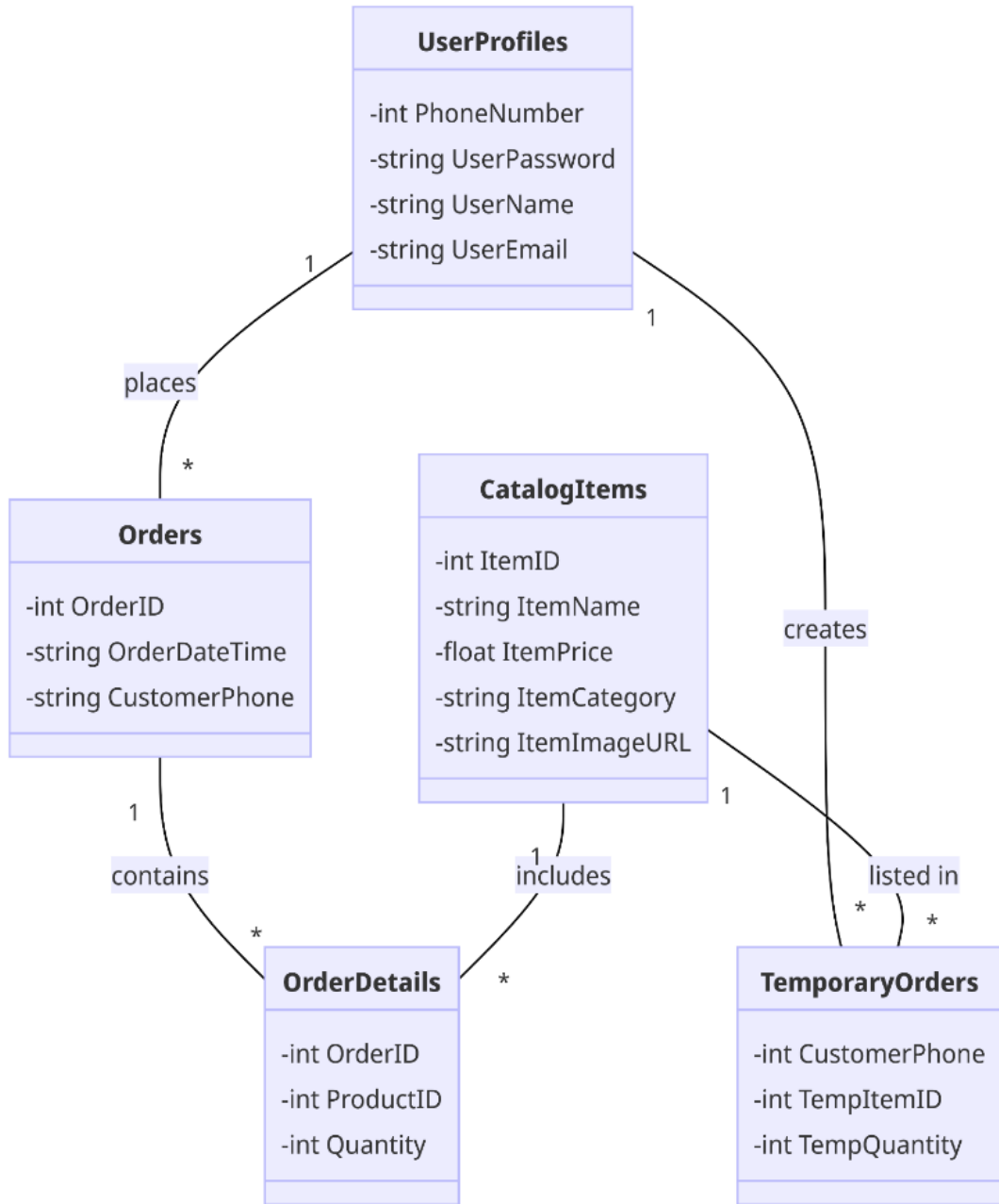


Рисунок 3.3 – Схема бази даних

Таблиця 3.1 "UserProfiles" в базі даних зберігає інформацію, необхідну для ідентифікації та аутентифікації користувачів в системі. Ось що робить кожен стовпець у цій таблиці:

- **PhoneNumber (int)**. Зберігає номер телефону користувача. Це поле може використовуватися для верифікації користувача та як спосіб зв'язку;

- `UserPassword` (string). Містить пароль користувача для входу в систему. Це важливий елемент забезпечення безпеки, оскільки пароль повинен бути захищений та недоступний для несанкціонованого доступу;
- `UserName` (string). Повне ім'я користувача. Це поле використовується для персоналізації взаємодії з користувачем та може використовуватися в кореспонденції з ним;
- `UserEmail` (string). Електронна адреса користувача, яка використовується для реєстрації в системі. Це основний засіб для зв'язку з користувачем, а також для відновлення доступу до акаунту в разі забування паролю чи інших проблем із входом.

Ця таблиця є ключовою для роботи системи користувацьких акаунтів, оскільки вона забезпечує збереження всієї необхідної інформації про користувачів для їхньої ідентифікації та авторизації в системі.

Таблиця 3.1 – UserProfiles

Назва	Тип	Опис
PhoneNumber	int	Номер телефону користувача
UserPassword	string	Пароль користувача для входу в систему
UserName	string	Повне ім'я користувача
UserEmail	string	Електронна адреса користувача для реєстрації

Таблиця 3.2 "CatalogItems" в базі даних служить для зберігання інформації про товари, які доступні у каталозі. Ось що робить кожен стовпець у цій таблиці:

- `ItemID` (int). Унікальний ідентифікатор товару. Використовується для точного визначення товару в базі даних та системі;
- `ItemName` (string). Найменування товару. Це поле містить назву товару, яку бачать користувачі, і використовується для ідентифікації товару в каталозі;

- ItemPrice (float). Вартість товару. Це числове значення вказує ціну, за яку товар продається. Це важливо для обчислення вартості покупок;
- ItemCategory (string). Категорія, до якої належить товар. Допомогає в класифікації та організації товарів у каталозі, спрощуючи пошук та фільтрацію для користувачів;
- ItemImageUrl (string). URL зображення товару. Це поле містить посилання на зображення товару, яке використовується для візуального представлення товару в мобільному додатку або на вебсайті.

Ця таблиця є основою для функціонування каталогу товарів в системі, оскільки забезпечує збереження всієї необхідної інформації для відображення, управління та продажу товарів.

Таблиця 3.2 – CatalogItems

Назва	Тип	Опис
ItemID	int	Унікальний ідентифікатор товару
ItemName	string	Найменування товару
ItemPrice	float	Вартість товару
ItemCategory	string	Категорія, до якої належить товар
ItemImageURL	string	URL зображення товару

Таблиці 3.3 -3.5 є важливими для управління процесами замовлень в системі.

Таблиця 3.3– OrderDetails

Назва	Тип	Опис
OrderID	int	Номер конкретного замовлення
ProductID	int	Ідентифікатор конкретного товару
Quantity	int	Кількість замовлених одиниць товару

Таблиця 3.4 – Orders

Назва	Тип	Опис
OrderID	int	Унікальний номер замовлення
OrderDateTime	string	Дата і час оформлення замовлення
CustomerPhone	string	Телефонний номер клієнта, що робить замовлення

Таблиця 3.5 – TemporaryOrders

Назва	Тип	Опис
CustomerPhone	int	Номер телефону клієнта
TempItemID	int	Ідентифікатор товару в тимчасовому замовленні
TempQuantity	int	Кількість одиниць товару в тимчасовому замовленні

"OrderDetails" зберігає інформацію про кількість та вид товарів у кожному замовленні, вказуючи на специфікації покупки. "Orders" фіксує загальні дані про замовлення, включаючи час та контакти клієнта, що дозволяє систематично відслідковувати та обробляти покупки. "TemporaryOrders" використовується для управління тимчасовими замовленнями, що дає можливість коригувати або відмінити замовлення до їх остаточного оформлення, гарантуючи гнучкість у процесі покупок.

3.3 Інтерфейс користувача та адміністративна панель

Після запуску застосунку користувач потрапляє на сторінку входу, де необхідно вказати свій телефонний номер та пароль для доступу до основного Меню. Якщо у користувача ще немає профілю, він може перейти до створення акаунту, використовуючи кнопку «Зареєструватися». У випадку помилок при вводі інформації на сторінках входу чи реєстрації з'явиться сповіщення про некоректне введення даних (рис. 3.4).

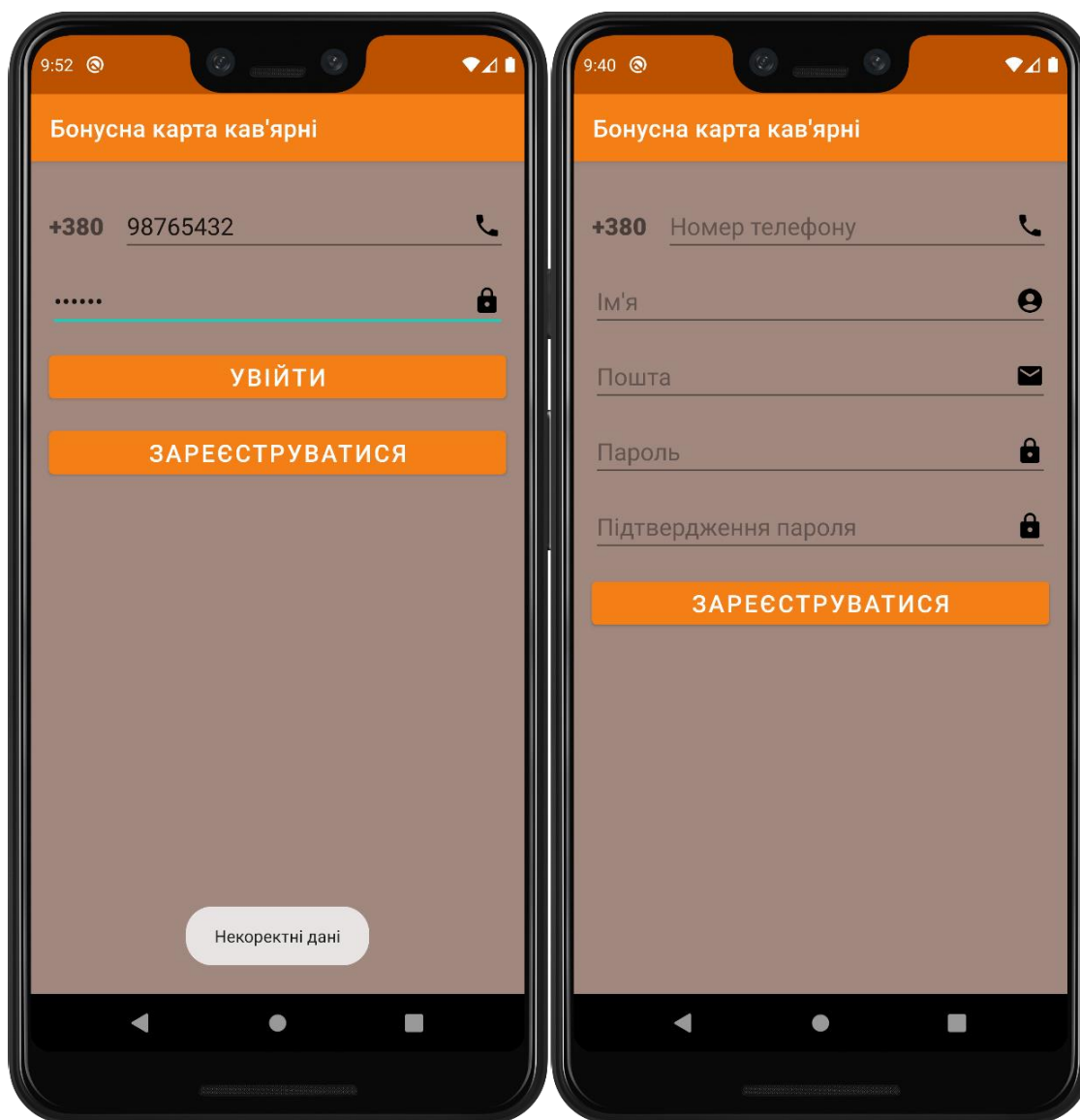


Рисунок 3.4 – Вхід користувача до системи

Коли користувач натискає «Увійти», йому пропонується перелік категорій, отриманий з бази даних. Вибравши категорію, застосунок переводить користувача на нову сторінку, де представлено асортимент товарів цієї категорії з назвами, цінами та можливістю додавання до кошика, інформація для чого також береться з бази даних.

Визначившись з вибором товару, користувач має натиснути на «Додати», після чого з'являється вікно для вказівки кількості товару. Далі користувачу показується список товарів у кошику (рис. 3.5).

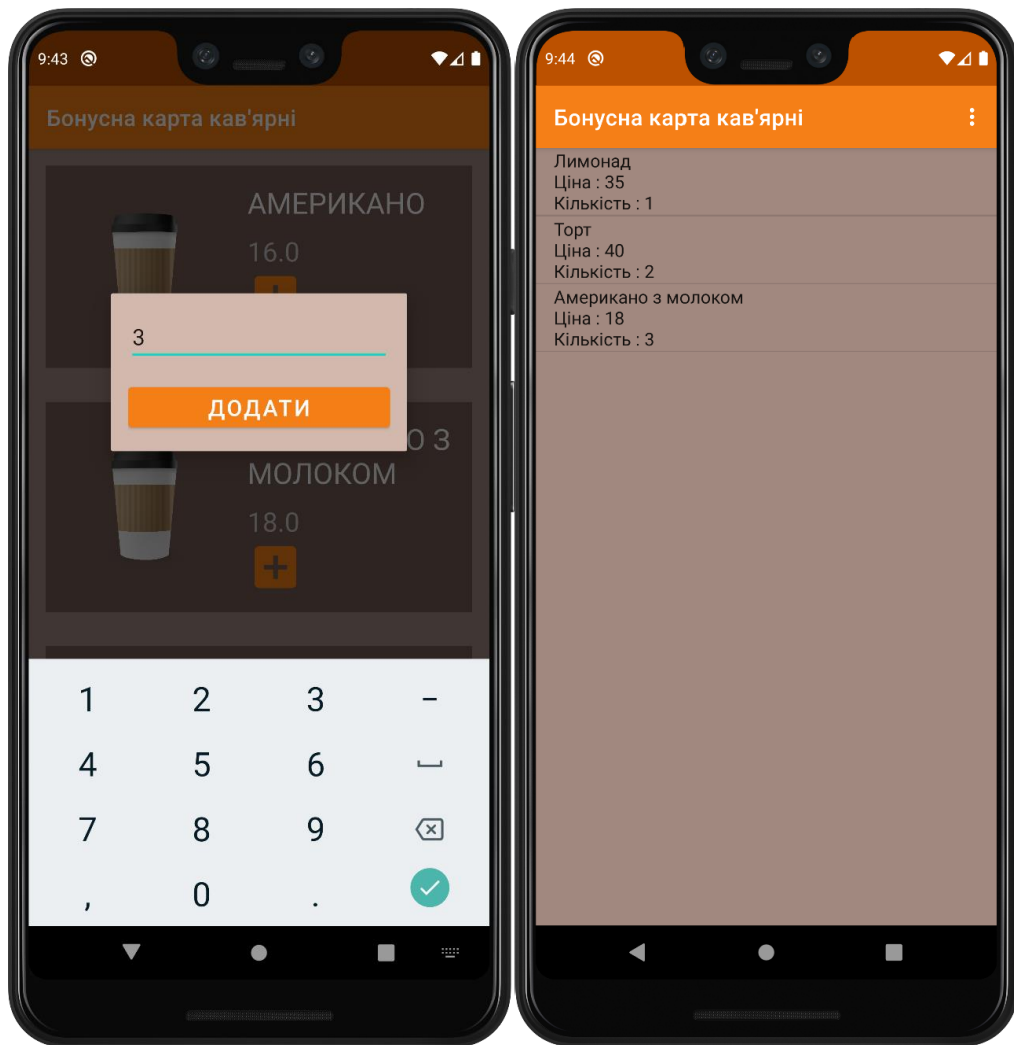


Рисунок 3.5 – Замовлення

У меню вибору користувач має визначитись з подальшими діями: обрати «Відмінити», щоб видалити замовлення з бази даних та повернутися до вибору категорій; натиснути «Підтвердити», щоб фіналізувати замовлення в базі даних із подальшим відображенням загальної вартості та номера замовлення, після чого застосунок переведе на сторінку з підтвердженням замовлення. Також присутня опція «Повернутися до меню» для переходу назад до основного Меню.

3.4 Висновки за розділом 3

У розділі описано реалізацію програмного забезпечення для Android-застосунку. Застосунок складається з декількох активностей, кожна з яких

відповідає за певний аспект взаємодії з користувачем: вхід/реєстрація, вибір категорій, вибір товарів, формування замовлення та перегляд бонусів.

Розроблений застосунок використовує архітектуру "клієнт-сервер", де мобільний застосунок виступає клієнтом, а взаємодія з базою даних відбувається через віртуальний сервер, налаштований на WampServer. Для обміну даними між клієнтом та сервером використовуються PHP-скрипти, які забезпечують необхідну функціональність (авторизація, реєстрація, отримання даних тощо).

База даних, реалізована на MySQL, забезпечує зберігання та управління даними про користувачів, товари, категорії та замовлення. Схема бази даних спроектована відповідно до вимог додатку з використанням нормалізації та визначенням ключових полів.

Інтерфейс користувача розроблений з урахуванням зручності та простоти використання, забезпечуючи логічний потік дій для формування замовлення. Адміністративна панель не описана, але можна припустити наявність засобів управління товарами та замовленнями через базу даних.

4 ТЕСТУВАННЯ ТА ВПРОВАДЖЕННЯ

4.1 Стратегія та методи тестування

Для забезпечення якості та надійності розробленого програмного забезпечення бонусної системи кав'ярні необхідно провести ретельне тестування. Стратегія тестування включатиме в себе різні методи та рівні перевірки функціональності, продуктивності та безпеки системи.

Модульне тестування (Unit Testing):

- мета: перевірка коректності роботи окремих модулів та компонентів системи;

- методи: написання автоматизованих тестів для кожного модуля, використовуючи фреймворки для тестування, такі як JUnit для Android та PHPUnit для серверної частини;

- очікувані результати: виявлення та виправлення помилок у логіці роботи окремих модулів, забезпечення стабільності базових компонентів системи.

Інтеграційне тестування (Integration Testing):

- мета: перевірка взаємодії між різними модулями та компонентами системи, а також інтеграції з зовнішніми сервісами (наприклад, база даних);

- методи: розробка тестових сценаріїв, які імітують реальні умови використання системи та перевіряють коректність обміну даними між компонентами;

- очікувані результати: виявлення проблем у взаємодії модулів, забезпечення безперебійної роботи системи в цілому.

Функціональне тестування (Functional Testing):

- мета: перевірка відповідності функціональних можливостей системи заявленим вимогам та очікуванням користувачів;

- методи: ручне тестування основних сценаріїв використання додатку (реєстрація, авторизація, вибір товарів, формування замовлення тощо), а також граничних умов та виняткових ситуацій;

- очікувані результати: підтвердження коректної роботи всіх функцій системи, виявлення та усунення невідповідностей та помилок.

Тестування користувацького інтерфейсу (UI Testing):

- мета: перевірка зручності, інтуїтивності та консистентності користувацького інтерфейсу мобільного додатку;

- методи: ручне тестування UI на різних пристроях та версіях Android, перевірка відповідності дизайну, коректності відображення елементів та зручності навігації;

- очікувані результати: забезпечення високої якості користувацького досвіду, виявлення та усунення проблем з відображенням та взаємодією з елементами інтерфейсу.

Тестування продуктивності (Performance Testing):

- мета: оцінка швидкодії, стабільності та масштабованості системи під навантаженням;

- методи: використання інструментів для навантажувального тестування, таких як Apache JMeter, для симуляції великої кількості одночасних запитів до сервера та аналізу часу відповіді, використання ресурсів тощо;

- очікувані результати: виявлення потенційних проблем з продуктивністю, оптимізація швидкодії системи, забезпечення стабільної роботи під високим навантаженням.

Тестування безпеки (Security Testing):

- мета: перевірка захищеності системи від потенційних загроз та вразливостей;

- методи: аналіз безпеки мережевої взаємодії, перевірка шифрування даних, тестування на наявність розповсюджених вразливостей (наприклад, SQL ін'єкції, XSS), перевірка процесу аутентифікації та авторизації користувачів;

- очікувані результати: виявлення та усунення потенційних проблем безпеки, забезпечення конфіденційності та цілісності даних користувачів.

Для ефективного виконання тестування розроблено детальний план тестування з описом тестових випадків та критеріїв приймання для кожного рівня тестування (табл. 4.1).

Таблиця 4.1 – Методи тестування

Рівень тестування	Мета	Методи
Модульне тестування	Перевірка коректності роботи окремих модулів та компонентів системи	Написання автоматизованих тестів для кожного модуля з використанням фреймворків JUnit для Android та PHPUnit для серверної частини
Інтеграційне тестування	Перевірка взаємодії між модулями та компонентами системи, інтеграції з зовнішніми сервісами	Розробка тестових сценаріїв, які імітують реальні умови використання системи та перевіряють коректність обміну даними між компонентами
Функціональне тестування	Перевірка відповідності функціональних можливостей системи заявленим вимогам та очікуванням	Ручне тестування основних сценаріїв використання додатку, граничних умов та виняткових ситуацій
Тестування інтерфейсу	Перевірка зручності, інтуїтивності та консистентності користувацького інтерфейсу	Ручне тестування UI на різних пристроях та версіях Android, перевірка відповідності дизайну, коректності відображення елементів та зручності навігації
Тестування продуктивності	Оцінка швидкодії, стабільності та масштабованості системи під навантаженням	Використання інструментів для навантажувального тестування (Apache JMeter) для симуляції великої кількості запитів та аналізу часу відповіді, використання ресурсів
Тестування безпеки	Перевірка захищеності системи від потенційних загроз та вразливостей	Аналіз безпеки мережевої взаємодії, перевірка шифрування даних, тестування на типові вразливості, перевірка аутентифікації та авторизації користувачів

Після завершення тестування та усунення всіх критичних дефектів, система буде готова до розгортання та впровадження у реальному середовищі кав'ярні. При цьому, процес тестування повинен бути

безперервним і повторюватись після кожного оновлення чи внесення змін до програмного забезпечення бонусної системи.

4.2 Результати тестування та усунення недоліків

Під час виконання тестування бонусної системи кав'ярні було отримано наступні позитивні результати.

Модульне тестування:

- всі розроблені модульні тести для компонентів Android-додатку та серверної частини успішно пройдені;
- виявлено та виправлено незначні помилки в логіці обробки граничних умов.

Інтеграційне тестування:

- підтверджено коректність взаємодії між активностями та фрагментами Android-додатку;
- обмін даними між клієнтською та серверною частинами системи відбувається без помилок;
- інтеграція з базою даних MySQL функціонує належним чином, запити виконуються правильно.

Функціональне тестування:

- сценарії реєстрації та авторизації користувачів працюють коректно;
- функції вибору категорій, товарів та формування замовлення виконуються без збоїв;
- розрахунок бонусів та відображення історії транзакцій відповідають очікуваним результатам.

Тестування користувацького інтерфейсу:

- інтерфейс додатку коректно відображається на різних розмірах екрану та версіях Android;
- навігація між екранами є зручною та інтуїтивно зрозумілою;

- ведення даних користувачем обробляється правильно, повідомлення про помилки відображаються коректно.

Тестування продуктивності:

- система демонструє стабільну роботу під високим навантаженням, обробляючи велику кількість одночасних запитів;
- час відгуку сервера та швидкість завантаження даних у мобільному додатку знаходяться в прийнятних межах;
- використання ресурсів сервера під час пікових навантажень не перевищує допустимих значень.

Тестування безпеки:

- з'єднання між клієнтом та сервером є захищеним завдяки використанню протоколу HTTPS;
- система аутентифікації та авторизації користувачів функціонує надійно, несанкціонований доступ неможливий;
- серверна частина стійка до атак SQL-ін'єкцій та XSS.

Результати тестування наведені в таблиці 4.2.

Незважаючи на успішне проходження більшості тестів, було виявлено кілька недоліків, які потребували усунення:

- виправлено помилку, пов'язану з некоректним розрахунком бонусів при певних умовах;
- оптимізовано запити до бази даних для зменшення часу відгуку сервера під високим навантаженням;
- додано додаткову валідацію вхідних даних на серверній частині для запобігання потенційним атакам.

Таблиця 4.2 – Тестування додатку

Функція	Результат
Реєстрація нового користувача	+
Авторизація користувача	+
Вибір категорій товарів	+
Відображення списку товарів обраної категорії	+
Додавання товару до кошика	+
Зміна кількості товару в кошику	+
Видалення товару з кошика	+
Оформлення замовлення	+
Відображення історії замовлень користувача	+
Нарахування бонусів за замовлення	+
Відображення балансу бонусного рахунку	+
Використання бонусів при оплаті замовлення	+
Відображення акційних пропозицій	+
Можливість зв'язку з службою підтримки через застосунок	+
Збереження даних користувача при виході з додатку	+
Коректне відображення елементів інтерфейсу на різних екранах	+
Стабільна робота додатку при великому навантаженні	+
Захищеність з'єднання між клієнтом та сервером	+
Стійкість серверної частини до SQL-ін'єкцій	+
Обробка некоректного введення даних користувачем	-

Після внесення відповідних виправлень, повторне тестування підтвердило, що виявлені недоліки були успішно усунені, і система відповідає всім заявленим вимогам.

4.3 Інструкції з впровадження та експлуатації системи

Для успішного впровадження та експлуатації бонусної системи кав'ярні необхідно виконати ряд кроків. Перш за все, потрібно підготувати серверну частину, розгорнувши вебсервер (наприклад, Apache) та встановивши необхідне програмне забезпечення (PHP, MySQL). Далі слід

імпортувати структуру бази даних та тестові дані з наданого SQL-файлу, розмістити PHP-скрипти на вебсервері та налаштувати з'єднання з базою даних. Для перевірки коректності роботи серверної частини необхідно протестувати API-запити.

Наступним кроком є розгортання мобільного додатку. Для Android потрібно згенерувати APK-файл додатку та розмістити його на надійному сервері для завантаження. Важливо забезпечити можливість встановлення додатку на пристрої співробітників та клієнтів кав'ярні, а також налаштувати застосунок для підключення до серверної частини, вказавши відповідну URL-адресу.

Для ефективного використання бонусної системи необхідно провести навчання персоналу. Слід організувати тренінги для співробітників кав'ярні щодо використання мобільного додатку та управління бонусною системою, забезпечити їх інструкціями та довідковими матеріалами для вирішення типових питань та проблем. Також важливо призначити відповідальних осіб для підтримки та адміністрування системи.

Інформування клієнтів про бонусну програму є ключовим фактором її успіху. Необхідно розмістити інформацію про бонусну програму на вебсайті кав'ярні та безпосередньо у закладі, провести рекламну кампанію для залучення клієнтів до участі в програмі. Персонал кав'ярні повинен бути готовим до надання клієнтам детальної інформації про переваги та умови бонусної системи, а також допомагати з встановленням та використанням мобільного додатку.

Після впровадження системи важливо забезпечити її постійну підтримку та оновлення. Слід регулярно проводити моніторинг роботи системи, збирати відгуки від користувачів та оперативно вирішувати виявлені проблеми. За необхідності, потрібно вносити зміни та доповнення до функціональності системи, щоб підтримувати її актуальність та відповідність потребам бізнесу.

Дотримання цих інструкцій дозволить успішно впровадити та ефективно експлуатувати бонусну систему кав'ярні, що сприятиме підвищенню лояльності клієнтів та зростанню прибутковості закладу.

4.4 Висновки за розділом 4

У четвертому розділі було розглянуто процес тестування та впровадження розробленого програмного забезпечення для бонусної системи кав'ярні.

Для забезпечення якості та надійності системи було розроблено комплексну стратегію тестування, яка включала різні методи та рівні перевірки функціональності, продуктивності та безпеки. Модульне тестування дозволило перевірити коректність роботи окремих компонентів системи, інтеграційне тестування підтвердило правильність взаємодії між модулями та зовнішніми сервісами, а функціональне тестування забезпечило відповідність програмного забезпечення заявленим вимогам та очікуванням користувачів. Крім того, було проведено тестування інтерфейсу, продуктивності та безпеки системи.

Результати тестування показали, що більшість функцій бонусної системи працює коректно та відповідає поставленим вимогам. Було виявлено та усунуто кілька недоліків, пов'язаних з розрахунком бонусів, оптимізацією запитів до бази даних та валідацією вхідних даних. Повторне тестування підтвердило успішне виправлення знайдених помилок.

Для успішного впровадження та експлуатації бонусної системи було розроблено детальні інструкції, які описують процес розгортання серверної та клієнтської частин, імпорту бази даних, налаштування з'єднання та забезпечення доступу до мобільного додатку. Також були надані рекомендації щодо навчання персоналу, інформування клієнтів та подальшої підтримки й оновлення системи.

Таким чином, ретельне тестування та дотримання інструкцій з впровадження дозволяють забезпечити стабільну та ефективну роботу бонусної системи кав'ярні, що сприятиме підвищенню лояльності клієнтів та зростанню прибутковості закладу. Процес тестування та вдосконалення програмного забезпечення повинен бути безперервним, щоб гарантувати його актуальність та відповідність потребам бізнесу.

ВИСНОВКИ

У ході виконання дипломної кваліфікаційної роботи бакалавра було досягнуто поставленої мети - розроблено програмне забезпечення для бонусної системи кав'ярні, яке дозволяє ефективно управляти процесами нарахування та використання бонусних балів клієнтами, підвищуючи їх лояльність та стимулюючи повторні відвідування закладу.

Для досягнення цієї мети було проведено ґрунтовний аналіз предметної області та існуючих рішень, що дозволило визначити основні вимоги та функціональні можливості програмного забезпечення. На основі цього аналізу було обґрунтовано вибір клієнт-серверної архітектури з використанням мобільного додатку для Android, що забезпечує зручність та доступність для клієнтів кав'ярні.

У процесі проектування було розроблено детальну архітектуру системи, змодельовано ключові бізнес-процеси за допомогою діаграм UML та обрано відповідний стек технологій, що включає мову програмування Kotlin для розробки мобільного додатку, PHP для серверної частини та СКБД MySQL для зберігання даних.

Реалізація програмного забезпечення здійснювалася з дотриманням принципів якісного кодування та використанням сучасних методологій розробки. Було створено зручний та інтуїтивно зрозумілий інтерфейс користувача, який забезпечує легку навігацію та доступ до всіх необхідних функцій. Взаємодія між клієнтом та сервером реалізована за допомогою REST API, що дозволяє ефективно обмінюватися даними та забезпечує масштабованість системи.

Для забезпечення якості та надійності розробленого програмного забезпечення було проведено ретельне тестування з використанням різних методів, таких як модульне, інтеграційне, функціональне тестування, а також тестування інтерфейсу користувача, продуктивності та безпеки. Виявлені

недоліки були проаналізовані та успішно усунені, що підтверджено результатами повторного тестування.

Окрім цього, було розроблено детальні інструкції з впровадження та експлуатації бонусної системи, що охоплюють процеси розгортання серверної та клієнтської частин, налаштування бази даних, а також рекомендації щодо навчання персоналу та інформування клієнтів.

Підсумовуючи, можна стверджувати, що розроблене програмне забезпечення повністю відповідає поставленим вимогам та очікуванням, демонструючи високу якість, зручність використання та безпеку. Впровадження цієї бонусної системи дозволить кав'ярні ефективно управляти лояльністю клієнтів, збільшити кількість повторних відвідувань та, як наслідок, підвищити прибутковість закладу.

Подальші напрямки розвитку проекту можуть включати розширення функціональних можливостей системи, інтеграцію з додатковими сервісами та платіжними системами, а також адаптацію програмного забезпечення для інших платформ, таких як iOS. Крім того, важливо забезпечити регулярну підтримку та оновлення системи відповідно до змін у потребах бізнесу та технологічному ландшафті.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Ахметов К. С. Практика програмування під Android. Київ: Кудіц-Прес, 2019. 288 с.
2. Голощапов А. Л. Google Android: програмування для мобільних пристроїв. Київ: прес, 2021. 448 с.
3. Дейтел П., Дейтел Х., Уолд А. Android для розробників. 3-є вид. Київ: Друк, 2022. 512 с.
4. Пелевін В. А. Розробка додатків під Android. Харків: ДМК Пресс, 2020. 402 с.
5. Amelia, R., Garg, L., Montgomery, J. A Practical Approach of Mobile Application Testing on Android Platform. International Journal of Information Technology and Computer Science. 2021. Vol. 11, No. 10. P. 37-47. DOI: 10.5815/ijitcs.2019.10.05
6. Busold, C., Heuser, S., Rios, E., Sadeghi, A. A Secure Development Lifecycle for Android Applications. IEEE Transactions on Dependable and Secure Computing. 2021. Vol. 17, No. 5. P. 1042-1055. DOI: 10.1109/TDSC.2020.2971485
7. Epifani, M., Stirparo, P. Learning Android Forensics. Birmingham: Packt Publishing Ltd., 2022. 292 p.
8. Phillips, B., Hardy, B., Stewart, C. Android Programming: The Big Nerd Ranch Guide. 4th ed. Atlanta: Big Nerd Ranch, 2020. 624 p.
9. Zapata, B. C. Android Studio Essentials. Birmingham: Packt Publishing Ltd., 2021. 180 p.

ДОДАТОК А
Технічне завдання

Вступ

Дане технічне завдання визначає вимоги до розробки програмного забезпечення для бонусної системи кав'ярні, яке дозволить ефективно управляти процесами нарахування та використання бонусних балів клієнтами, підвищуючи їх лояльність та стимулюючи повторні відвідування закладу.

A.1 Підстава для розробки

Підставою для розробки є завдання на дипломну кваліфікаційну роботу на тему "Розробка програмного забезпечення для бонусної системи кав'ярні", затверджене наказом Національного університету "Запорізька політехніка" №168 від 24 травня 2024 р.

A.2 Призначення розробки

Програмне забезпечення призначене для автоматизації процесів управління бонусною системою кав'ярні, включаючи реєстрацію клієнтів, нарахування бонусів за покупки, відстеження балансу бонусних рахунків та використання бонусів при оплаті замовлень.

A.3 Основні вимоги до програми, що розробляється

A.3.1 Вимоги до функціональних характеристик

Мобільний застосунок для клієнтів:

- реєстрація та авторизація користувачів;
- перегляд балансу бонусного рахунку та історії транзакцій;
- сканування QR-коду або введення коду при оплаті для нарахування бонусів;

- використання бонусів при оформленні замовлення;
- отримання персоналізованих пропозицій та акцій.

A.3.2 Вимоги до інтерфейсу програми

Інтерфейс користувача передбачається:

- інтуїтивно зрозумілий та привабливий дизайн мобільного додатку;
- адаптивність інтерфейсу до різних розмірів екрану смартфонів;
- зручна навігація та логічне розташування елементів керування;
- відображення корпоративних кольорів та логотипу кав'ярні.

A.3.3 Вимоги до надійності

Забезпечення безпеки та конфіденційності даних користувачів. Стабільна робота системи під високим навантаженням. Регулярне резервне копіювання даних. Захист від несанкціонованого доступу та потенційних вразливостей.

A.3.4 Умови експлуатації

Мобільний застосунок розробляється для платформи Android (версія 6.0 і вище). Серверна частина розгортається на надійному хостингу з підтримкою необхідних технологій.

A.3.5 Вимоги до складу та параметрів технічний засобів

Сервер з достатньою обчислювальною потужністю та об'ємом пам'яті. Система управління базами даних MySQL для зберігання даних. Мобільні пристрої користувачів з ОС Android та доступом до мережі Інтернет.

А.3.6 Вимоги до маркування і пакування

Програмне забезпечення може поширюватись у вигляді вихідного коду або виконуваних файлів. На пакуванні має бути назва "Розробка програмного забезпечення для бонусної системи кав'ярні".

ДОДАТОК Б
Текст програми

```

import android.content.Intent
import android.os.Bundle
import android.widget.Button
import android.widget.EditText
import androidx.appcompat.app.AppCompatActivity

class StartActivity : AppCompatActivity() {
    private lateinit var usernameEditText: EditText
    private lateinit var passwordEditText: EditText
    private lateinit var loginButton: Button
    private lateinit var signUpButton: Button

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_start)

        usernameEditText =
findViewById(R.id.usernameEditText)
        passwordEditText =
findViewById(R.id.passwordEditText)
        loginButton = findViewById(R.id.loginButton)
        signUpButton = findViewById(R.id.signUpButton)

        loginButton.setOnClickListener {
            val username = usernameEditText.text.toString()
            val password = passwordEditText.text.toString()
<?php
// Підключення до бази даних
require 'db_connection.php';

$username = $_POST['username'] ?? '';
$password = $_POST['password'] ?? '';

if (!empty($username) && !empty($password)) {
    // Запит для перевірки існування користувача

```

```

        $sql = "SELECT * FROM users WHERE username = ? AND
password = ?";

        $stmt = $conn->prepare($sql);
        $stmt->bind_param("ss", $username, $password);
        $stmt->execute();
        $result = $stmt->get_result();

        if ($result->num_rows > 0) {
            // Користувач знайдений
            echo "success";
        } else {
            // Користувач не знайдений
            echo "fail";
        }
    } else {
        echo "empty";
    }
?>

    }

import android.os.Bundle
import android.widget.Button
import android.widget.EditText
import androidx.appcompat.app.AppCompatActivity

class SignUpActivity : AppCompatActivity() {
    private lateinit var usernameEditText: EditText
    private lateinit var passwordEditText: EditText
    private lateinit var signUpButton: Button

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_sign_up)

        usernameEditText

        findViewById(R.id.usernameEditText)

```

```

        passwordEditText
findViewById(R.id.passwordEditText)
        signUpButton = findViewById(R.id.signUpButton)

        signUpButton.setOnClickListener {
            val username = usernameEditText.text.toString()
            val password = passwordEditText.text.toString()
<?php
// Підключення до бази даних
require 'db_connection.php';

$username = $_POST['username'] ?? '';
$password = $_POST['password'] ?? '';

if (!empty($username) && !empty($password)) {
    // Додавання нового користувача
    $sql = "INSERT INTO users (username, password) VALUES
(? , ?)";
    $stmt = $conn->prepare($sql);
    $stmt->bind_param("ss", $username, $password);

    if ($stmt->execute()) {
        echo "success";
    } else {
        echo "error";
    }
} else {
    echo "empty";
}
?>
        }
    }

        signUpButton.setOnClickListener {
            val            intent            =            Intent(this,
SignUpActivity::class.java)

```

```

        startActivity(intent)
    }
}

import android.content.Intent
import android.os.Bundle
import android.widget.ArrayAdapter
import android.widget.ListView
import androidx.appcompat.app.AppCompatActivity

class CategorySelectActivity : AppCompatActivity() {
    private lateinit var categoryListView: ListView

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_category_select)

        categoryListView =
findViewById(R.id.categoryListView)

<?php
// Підключення до бази даних
require 'db_connection.php';

// Отримання категорій
$sql = "SELECT * FROM categories";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
    $categories = array();
    while ($row = $result->fetch_assoc()) {
        $categories[] = $row;
    }
    echo json_encode($categories);
} else {
    echo "no_categories";
}

```

```

    }
    ?>
        val adapter = ArrayAdapter(this,
android.R.layout.simple_list_item_1, categories)
        categoryListView.adapter = adapter

        categoryListView.setOnItemClickListener { _, _,
position, _ ->
            val intent = Intent(this,
ItemSelectActivity::class.java)
            intent.putExtra("category",
categories[position])
            startActivity(intent)
        }
    }
}

import android.os.Bundle
import android.widget.ListView
import androidx.appcompat.app.AppCompatActivity

class ItemSelectActivity : AppCompatActivity() {
    private lateinit var itemListView: ListView

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_item_select)

        itemListView = findViewById(R.id.itemListView)

        val category = intent.getStringExtra("category")
<?php
// Підключення до бази даних
require 'db_connection.php';

$category_id = $_POST['category_id'] ?? '';

```

```

if (!empty($category_id)) {
    // Отримання товарів за категорією
    $sql = "SELECT * FROM items WHERE category_id = ?";
    $stmt = $conn->prepare($sql);
    $stmt->bind_param("i", $category_id);
    $stmt->execute();
    $result = $stmt->get_result();

    if ($result->num_rows > 0) {
        $items = array();
        while ($row = $result->fetch_assoc()) {
            $items[] = $row;
        }
        echo json_encode($items);
    } else {
        echo "no_items";
    }
} else {
    echo "empty";
}
?>

    val adapter = ItemAdapter(this, items)
    itemListView.adapter = adapter

    itemListView.setOnItemClickListener { _, _,
position, _ ->
    }

}

import android.os.Bundle
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.Button
import android.widget.EditText

```

```

import androidx.fragment.app.Fragment

class QuantityFragment : Fragment() {
    private lateinit var quantityEditText: EditText
    private lateinit var confirmButton: Button

    override fun onCreateView(inflater: LayoutInflater,
container: ViewGroup?, savedInstanceState: Bundle?): View? {
        val view =
inflater.inflate(R.layout.fragment_quantity, container, false)

        quantityEditText =
view.findViewById(R.id.quantityEditText)
        confirmButton =
view.findViewById(R.id.confirmButton)

        confirmButton.setOnClickListener {
            val quantity =
quantityEditText.text.toString().toInt()

            <?php
            // Підключення до бази даних
            require 'db_connection.php';

            $user_id = $_POST['user_id'] ?? '';
            $item_id = $_POST['item_id'] ?? '';
            $quantity = $_POST['quantity'] ?? '';

            if (!empty($user_id) && !empty($item_id) &&
!empty($quantity)) {
                // Додавання тимчасового замовлення
                $sql = "INSERT INTO temp_order (user_id, item_id,
quantity) VALUES (?, ?, ?)";
                $stmt = $conn->prepare($sql);
                $stmt->bind_param("iii", $user_id, $item_id,
$quantity);

```

```

        if ($stmt->execute()) {
            echo "success";
        } else {
            echo "error";
        }
    } else {
        echo "empty";
    }
?>

    }

    return view
}
}

import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity

class OrderReviewActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_order_review)
    }
}

import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity

class FinalSummaryActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_final_summary)
    }
}

```

ДОДАТОК В
Слайди презентації

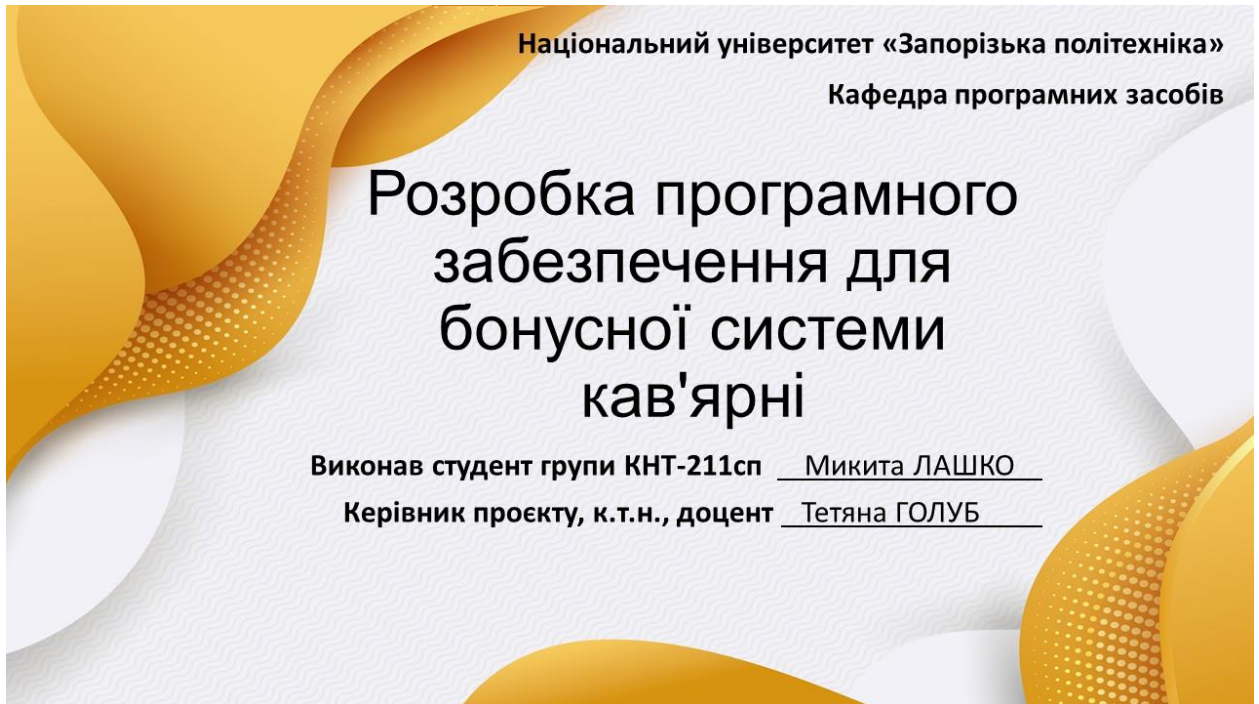


Рисунок В.1 – Слайд 1

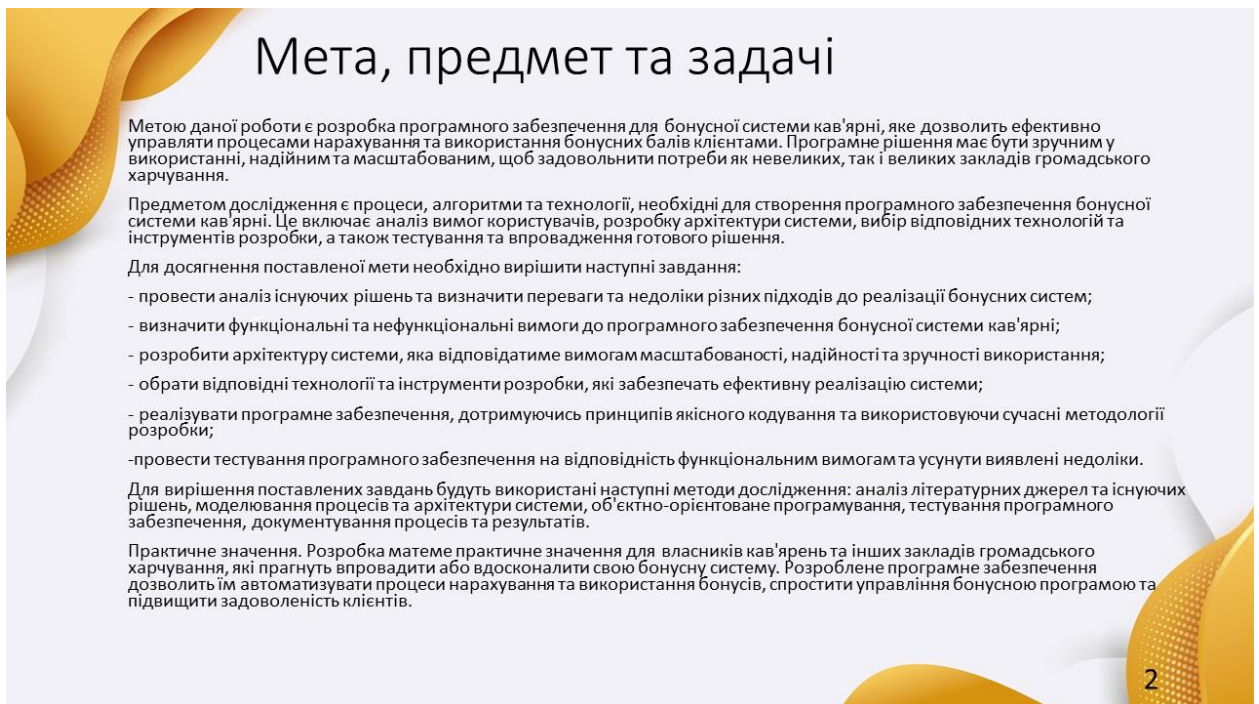


Рисунок В.2 – Слайд 2

Порівняння типів бонусних систем

Тип бонусної системи	Переваги	Недоліки
Програми лояльності на основі карток	- прості у використанні для клієнтів; - фізичний носій бонусів;	- необхідність виготовлення та розподілу карток; - ризик втрати або пошкодження карток;
Програми лояльності на основі мобільних додатків	- зручність для клієнтів; - збір даних про клієнтів; - легкість інформування про акції;	- не всі клієнти можуть встановити додаток - додаткові витрати на розробку додатку;
Системи знижок за повторні відвідування	- проста концепція для розуміння клієнтів; - мінімальні витрати на впровадження;	- обмежені можливості для персоналізації; - складність відстеження повторних відвідувань;
Програми, орієнтовані на події	- стимулювання відвідувань у певні періоди; - можливість залучення нових клієнтів;	- короткостроковий ефект - складність планування та впровадження;
Програми партнерства	- розширення переваг для клієнтів; - взаємний обмін клієнтськими базами;	- необхідність налагодження співпраці з партнерами; - ризики, пов'язані з репутацією партнерів.

3

Рисунок В.3 – Слайд 3

Програмні рішення для реалізації бонусних систем

Рішення	Переваги	Недоліки
Спеціалізовані платформи для управління лояльністю	- широкий функціонал для керування програмами лояльності; - інтеграція з різними системами лояльності; - аналітика та звітність;	- високі витрати на ліцензії та підтримку; - складність інтеграції з існуючими системами; - обмежена гнучкість налаштувань;
Модулі лояльності в POS-системах	- безпосередня інтеграція з касовими системами; - відносно низькі витрати на впровадження;	- обмежений функціонал управління програмою; - відсутність веб-інтерфейсу для клієнтів; - складність масштабування та оновлення;
Самостійна розробка на базі фреймворків/CMS	- повна гнучкість та налаштування під потреби; - немає прив'язки до зовнішніх постачальників; - можливість інтеграції з різними системами;	- високі початкові витрати на розробку; - необхідність підтримки та оновлення власними силами; - ризики забезпечення належної безпеки та продуктивності.

4

Рисунок В.4 – Слайд 4

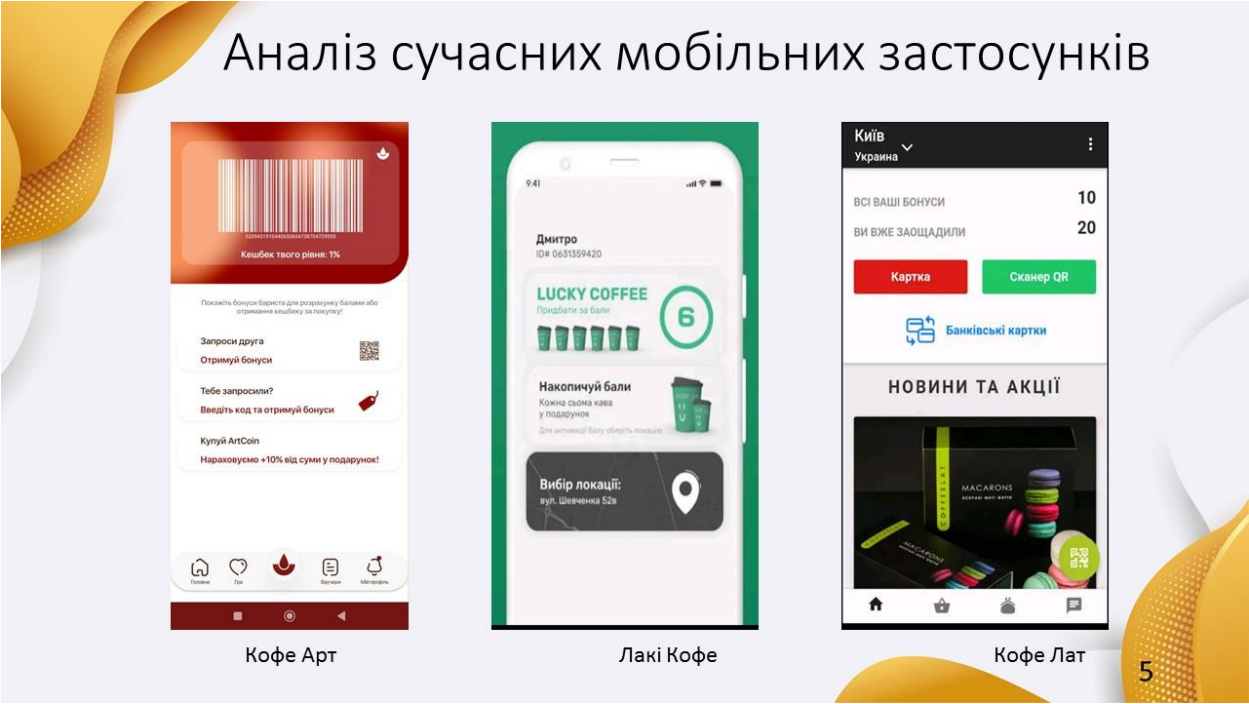


Рисунок В.5 – Слайд 5

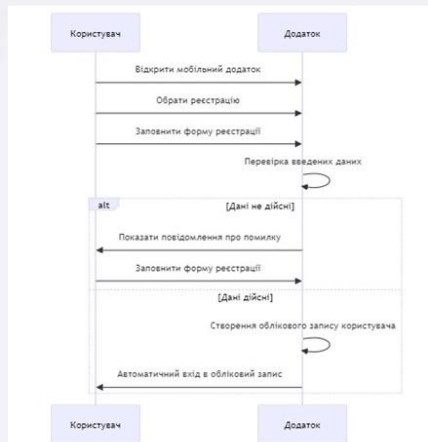
Порівняння застосунків

Застосунків	Переваги	Недоліки
Кофе Арт	Інтуїтивний інтерфейс, яскравий дизайн, простота використання, візуальна привабливість, легке сканування QR.	Потреба в масштабованості та безпеці даних, ризик збоїв при великій кількості запитів.
Лакі Кофе	Чистий дизайн, легка навігація, послідовне брендування, інтеграція з картою локацій.	Потенційні проблеми з користувачем на менших екранах, важливість стабільної роботи додатку під час пікових навантажень.
Кофе Лат	Функціональність, зручні кнопки для навігації, яскраві кольори, розділ "Новини та акції" для маркетингу.	Необхідність чіткого відображення інформації про бонуси, висока важливість безпеки даних та продуктивності.

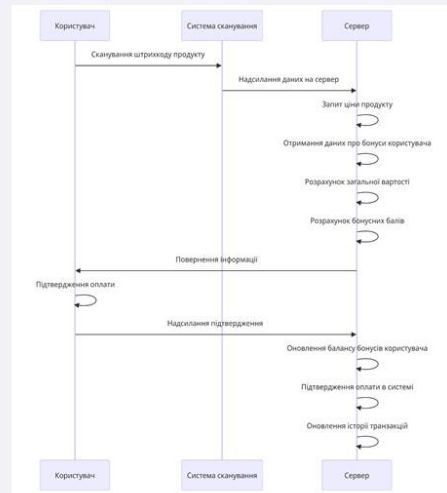
A yellow decorative element with the number '6' is visible on the right side.

Рисунок В.6 – Слайд 6

Моделювання додатку



Діаграма послідовності реєстрації нового користувача



Діаграма послідовності нарахування бонусів

7

Рисунок В.7 – Слайд 7

Вибір технологій розробки

Таблиця 1 – Порівняння додатків iOS та Android

Аспект	iOS	Android
Мови програмування та інструменти	Swift або Objective-C, інтегроване середовище розробки Xcode	Java або Kotlin, інтегроване середовище розробки Android Studio
Доступ до системи та налаштування	Обмежений доступ до системи для підвищення безпеки	Відкритий доступ, більш широкі можливості налаштування
Вимоги до дизайну інтерфейсу	Строге дотримання вказівок Apple Human Interface Guidelines	Гнучкі вказівки Google Material Design
Вимоги до публікації в магазині	Суворий процес перевірки в App Store	Більш м'який процес перевірки в Google Play
Заходи безпеки	Високі, завдяки суворому контролю за додатками та конфіденційністю	Можуть бути менші через відкриті природу екосистеми
Вартість розробки	Зазвичай вища через ретельне тестування та контроль якості	Зазвичай нижча через широкий спектр пристроїв та менш жорсткі вимоги
Частка ринку	Менша у порівнянні з Android, але користувачі частіше готові платити	Більша у порівнянні з iOS, з широким спектром пристроїв та цінових точок

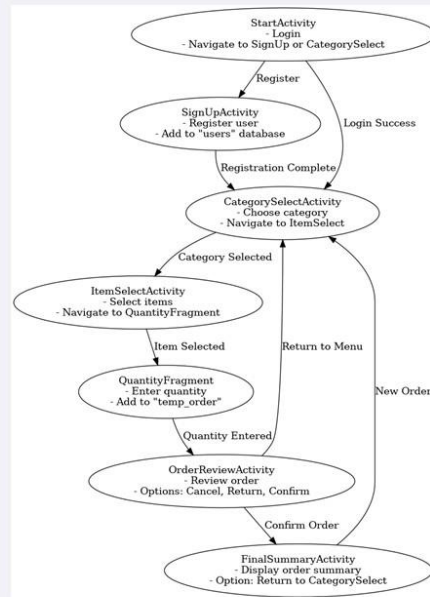
Таблиця 2 – Порівняння Java та Kotlin

Критерій	Java	Kotlin
Мова програмування	Старша, широко використовується для Android-розробки	Новіша, офіційно підтримується Google для Android-розробки
Синтаксис	Більш об'ємний і складний	Коротший і чистіший, спрощує читання та написання коду
Безпека використання null	Частіше стикається з помилками NullPointerException	Використовує безпечні від null типи, щоб уникнути NullPointerException
Сумісність з Java	Нативна сумісність, оскільки Kotlin може використовувати Java бібліотеки і навпаки	Повна сумісність, дозволяє легко інтегрувати Kotlin у існуючі Java проекти
Функціональне програмування	Обмежені можливості порівняно з Kotlin	Підтримує функціональні парадигми ефективніше, ніж Java
Корутини	Не підтримується нативно, потрібні додаткові бібліотеки	Підтримується нативно, спрощує асинхронне програмування
Спільнота та підтримка	Велика та досвідчена, але Kotlin набуває популярності	Швидко зростаюча з сильною підтримкою від Google і спільноти

8

Рисунок В.8 – Слайд 8

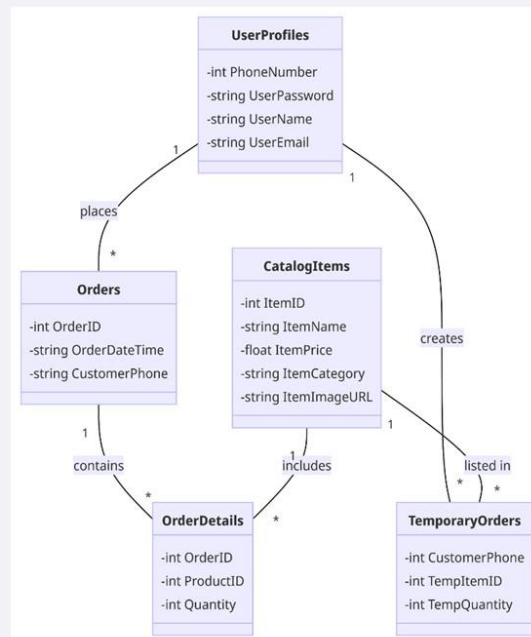
Робота програмних модулів додатку



9

Рисунок В.9 – Слайд 9

База даних



10

Рисунок В.10 – Слайд 10

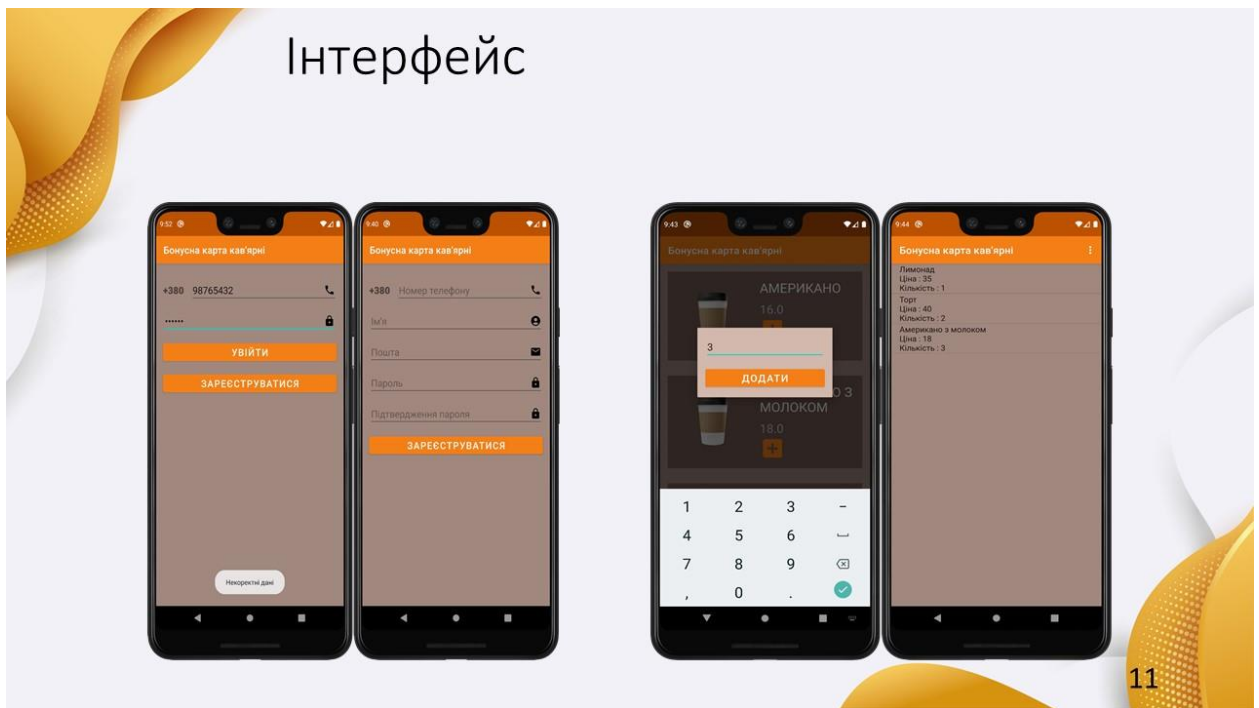


Рисунок В.11 – Слайд 11

Тестування

Функція	Результат
Реєстрація нового користувача	+
Авторизація користувача	+
Вибір категорій товарів	+
Відображення списку товарів обраної категорії	+
Додавання товару до кошика	+
Зміна кількості товару в кошику	+
Видалення товару з кошика	+
Оформлення замовлення	+
Відображення історії замовлень користувача	+
Нарахування бонусів за замовлення	+
Відображення балансу бонусного рахунку	+
Використання бонусів при оплаті замовлення	+
Відображення акційних пропозицій	+
Можливість зв'язку з службою підтримки через додаток	+
Збереження даних користувача при виході з додатку	+
Коректне відображення елементів інтерфейсу на різних екранах	+
Стабільна робота додатку при великому навантаженні	+
Захищеність з'єднання між клієнтом та сервером	+
Стійкість серверної частини до SQL-ін'єкцій	+
Обробка некоректного введення даних користувачем	-

Рисунок В.12 – Слайд 12

Висновки

У ході даної роботи було успішно досягнуто поставленої мети - розробки програмного забезпечення для бонусної системи кав'ярні, яке відповідає вимогам ефективного управління процесами нарахування та використання бонусних балів клієнтами. Розроблене рішення є зручним у використанні, надійним та масштабованим, що дозволяє задовольнити потреби як невеликих, так і великих закладів громадського харчування.

Для досягнення мети були успішно виконані всі поставлені завдання. Проведено аналіз існуючих рішень, визначено переваги та недоліки різних підходів до реалізації бонусних систем. На основі цього аналізу було визначено функціональні та нефункціональні вимоги до програмного забезпечення. Розроблено архітектуру системи, яка відповідає вимогам масштабованості, надійності та зручності використання. Обрано відповідні технології та інструменти розробки, що забезпечили ефективну реалізацію системи. Програмне забезпечення було реалізовано з дотриманням принципів якісного кодування та використанням сучасних методологій розробки. Проведено тестування програмного забезпечення на відповідність функціональним вимогам та усунуто виявлені недоліки.

Таким чином, розроблене програмне забезпечення для бонусної системи кав'ярні має практичне значення для власників закладів громадського харчування, дозволяючи їм автоматизувати процеси нарахування та використання бонусів, спростити управління бонусною програмою та підвищити задоволеність клієнтів.

Рисунок В.13 – Слайд 13